РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖ|БЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра теории вероятностей и кибербезопасности

Лабораторная работа № 3 | Управляющие структуры

Студент: Абд эль хай Мохамад
Номер: 1032215163
Група: НПИбд-01-21

Москва 2025

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

Циклы while и for

In [1]:
```
n=0
while n < 10
    n += 1
    print(" ",n)
end
```

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

In [2]:
```
myfreinds = ["Vova","Andrey","Naimen"]
i =1
while i<=length(myfreinds)
    freind = myfreinds[i]
    println("Hi $freind ")
    i += 1
end
```

```
Hi Vova
Hi Andrey
Hi Naimen
```

In [6]:
```
for i in myfreinds
    println(" Hi $i ")
end
```

```
Hi Vova
Hi Andrey
Hi Naimen
```

In [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
A
```

Out[2]:
```
5×5 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
```

In [3]:
```
B = fill(0,(m,n))

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

Out[3]:
```
5×5 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
```

In [4]:
```
C = [ i+j for i in 1:m, j in 1:n]
C
```

Out[4]:
```
5×5 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
```

Условные выражения

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

In [6]:
```
N = 5

if ( N%3 == 0)&&( N%5 == 0)
    println("FizzBuzz")
elseif N%3 == 0
    println("Fizz")
elseif N%5 == 0
```

```
    println("Buzz")
else
    prinln(N)
end
```

```
Buzz
```

In [7]:
```
x = 5
y = 10
(x > y) ? x : y
```

Out[7]:
```
10
```

Функции

Julia дает нам несколько разных способов написать функцию. Первый требует ключевых слов function и end:

In [11]:
```
function sayhi(name)
    println("Hi $name, it's great to see you!")
end

sayhi("Mohamad")
```

```
Hi Mohamad, it's great to see you!
```

In [12]:
```
function f(x)
    x^2
end

f(3)
```

Out[12]:
```
9
```

Также можно объявить вышеперечисленные функции как анонимные

In [13]:
```
sayhi3 = name -> println("Hi $name, it's great to see you!")
sayhi3("Bernard")
```

```
Hi Bernard, it's great to see you!
```

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

In [15]:
```
v = [3,5, 2]
sort(v)
println(v)
sort!(v)
v
```

```
[3, 5, 2]
```

Out[15]:
```
3-element Vector{Int64}:
 2
 3
 5
```

Функция broadcast() будет пытаться привести все объекты к общему измерению, map() будет напрямую применять данную функцию поэлементно.

In [16]:
```
map(f, [1, 2, 3])

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

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

In [17]:
```
A = [i + 3*j for j in 0:2, i in 1:3]
```

Out[17]:
```
3×3 Matrix{Int64}:
 1  2  3
 4  5  6
 7  8  9
```

In [18]:
```
f(A)
```

Out[18]:
```
3×3 Matrix{Int64}:
  30   36   42
  66   81   96
 102  126  150
```

In [19]:
```
B= f.(A)
```

Out[19]:
```
3×3 Matrix{Int64}:
  1   4   9
 16  25  36
 49  64  81
```

Сторонние библиотеки (пакеты) в Julia

In [20]:
```
import Pkg
Pkg.add("Example")
```

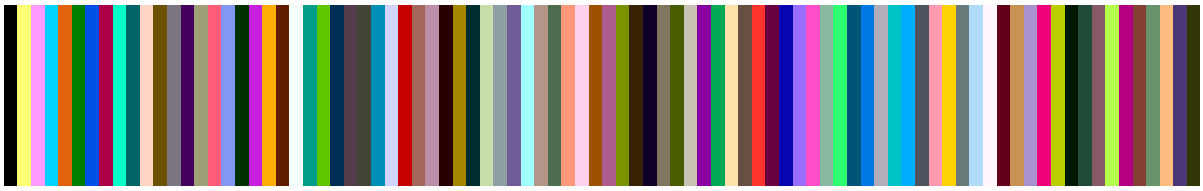
```
    Updating registry at `~/.julia/registries/General.toml`
   Resolving package versions...
   Installed Example ─ v0.5.5
    Updating `~/.julia/environments/v1.11/Project.toml`
  [7876af07] + Example v0.5.5
    Updating `~/.julia/environments/v1.11/Manifest.toml`
  [7876af07] + Example v0.5.5
Precompiling project...
   1524.9 ms  ✓ Example
  1 dependency successfully precompiled in 3 seconds. 43 already precompile
d.
```

In [21]:
```
Pkg.add("Colors")
using Colors
```

```
   Resolving package versions...
   Installed Reexport ──────────── v1.2.2
   Installed FixedPointNumbers ─ v0.8.5
   Installed ColorTypes ────────── v0.12.0
   Installed Statistics ────────── v1.11.1
   Installed Colors ────────────── v0.13.0
    Updating `~/.julia/environments/v1.11/Project.toml`
  [5ae59095] + Colors v0.13.0
    Updating `~/.julia/environments/v1.11/Manifest.toml`
  [3da002f7] + ColorTypes v0.12.0
  [5ae59095] + Colors v0.13.0
  [53c48c17] + FixedPointNumbers v0.8.5
  [189a3867] + Reexport v1.2.2
  [10745b16] + Statistics v1.11.1
  [37e2e46d] + LinearAlgebra v1.11.0
  [e66e0078] + CompilerSupportLibraries_jll v1.1.1+0
  [4536629a] + OpenBLAS_jll v0.3.27+1
  [8e850b90] + libblastrampoline_jll v5.11.0+0
Precompiling project...
    744.2 ms  ✓ Reexport
    917.6 ms  ✓ Statistics
   3319.8 ms  ✓ FixedPointNumbers
   2029.0 ms  ✓ ColorTypes
    973.6 ms  ✓ ColorTypes → StyledStringsExt
   7025.6 ms  ✓ Colors
  6 dependencies successfully precompiled in 14 seconds. 45 already precompi
led.
```

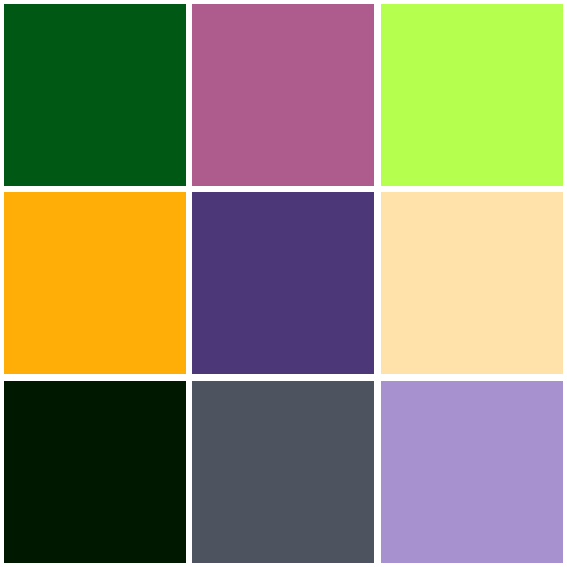
In [22]:
```
palette = distinguishable_colors(100)
```

Out[22]:



In [24]:
```
rand(palette, 3, 3)
```

Out[24]:



Задания для самостоятельного выполнения

1. Используя циклы while и for:
   - выведите на экран целые числа от 1 до 100 и напечатайте их квадраты;
   - создайте словарь squares, который будет содержать целые числа в качестве ключей и квадраты в качестве их пар-значений;
   - создайте массив squares_arr, содержащий квадраты всех чисел от 1 до 100.

In [26]:
```
for i in 1:100
    print(" ",i)
end
```

```
 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 5
4 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
```

In [1]:
```
i =1
while i <= 100
    print(" ",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
529 576 625 676 729 784 841 900 961 1024 1089 1156 1225 1296 1369 1444 1521
1600 1681 1764 1849 1936 2025 2116 2209 2304 2401 2500 2601 2704 2809 2916 3
025 3136 3249 3364 3481 3600 3721 3844 3969 4096 4225 4356 4489 4624 4761 49
00 5041 5184 5329 5476 5625 5776 5929 6084 6241 6400 6561 6724 6889 7056 722
5 7396 7569 7744 7921 8100 8281 8464 8649 8836 9025 9216 9409 9604 9801 1000
0
```

In [5]:
```
squares = Dict{Int64, Int64}()
# занесем значения в словарь
for i in 1:100
```

```
    push!(squares, i => i^2)
end
print(pairs(squares))
```

```
Dict(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 => 810
0, 4 => 16, 13 => 169, 54 => 2916, 63 => 3969, 86 => 7396, 91 => 8281, 62 =>
3844, 58 => 3364, 52 => 2704, 12 => 144, 28 => 784, 75 => 5625, 23 => 529, 9
2 => 8464, 41 => 1681, 43 => 1849, 11 => 121, 36 => 1296, 68 => 4624, 69 =>
4761, 98 => 9604, 82 => 6724, 85 => 7225, 39 => 1521, 84 => 7056, 77 => 592
9, 7 => 49, 25 => 625, 95 => 9025, 71 => 5041, 66 => 4356, 76 => 5776, 34 =>
1156, 50 => 2500, 59 => 3481, 93 => 8649, 2 => 4, 10 => 100, 18 => 324, 26 =
> 676, 27 => 729, 42 => 1764, 87 => 7569, 100 => 10000, 79 => 6241, 16 => 25
6, 20 => 400, 81 => 6561, 19 => 361, 49 => 2401, 44 => 1936, 9 => 81, 31 =>
961, 74 => 5476, 61 => 3721, 29 => 841, 94 => 8836, 46 => 2116, 57 => 3249,
70 => 4900, 21 => 441, 38 => 1444, 88 => 7744, 78 => 6084, 72 => 5184, 24 =>
576, 8 => 64, 17 => 289, 37 => 1369, 1 => 1, 53 => 2809, 22 => 484, 47 => 22
09, 83 => 6889, 99 => 9801, 89 => 7921, 14 => 196, 3 => 9, 80 => 6400, 96 =>
9216, 51 => 2601, 33 => 1089, 40 => 1600, 48 => 2304, 15 => 225, 65 => 4225,
97 => 9409)
```

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

In [6]:
```
N = 5
if N % 2 == 0
    println("Четное число")
else
    println("Нечетное число")
end
```

```
Нечетное число
```

In [7]:
```
N = 4
(N % 2 == 0) ? println("Четное число") : println("Нечетное число")
```

```
Четное число
```

3. Напишите функцию add_one, которая добавляет 1 к своему входу.

In [8]:
```
function add_one(X)
    X + 1
end
add_one(5)
```

Out[8]: 6

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

In [9]:
```
X = fill(1, 4 * 4)
Y = collect(0:(length(X) - 1))
```

```
X = reshape(map(+, X, Y), (4, 4))
```

Out[9]:
```
4×4 Matrix{Int64}:
 1  5   9  13
 2  6  10  14
 3  7  11  15
 4  8  12  16
```

5. Задайте матрицу A

In [10]:
```
A = [1 1 3; 5 2 6; -2 -1 -3]
```

Out[10]:
```
3×3 Matrix{Int64}:
  1   1   3
  5   2   6
 -2  -1  -3
```

In [11]:
```
A^3
```

Out[11]:
```
3×3 Matrix{Int64}:
 0  0  0
 0  0  0
 0  0  0
```

In [12]:
```
for i in 7:1:9
    A[i] += A[i-3]
end
A
```

Out[12]:
```
3×3 Matrix{Int64}:
  1   1   4
  5   2   8
 -2  -1  -4
```

6. Создайте матрицу B с элементами $Bi1 = 10$, $Bi2 = -10$, $Bi3 = 10$, $i = 1, 2, \ldots, 15$.
Вычислите матрицу $C = (B\text{^}T)B$.

In [13]:
```
# объявим матрицу B
B = Array{Int32, 2}(undef, 15, 3)

# заполним матрицу в соответствии с заданием
for i in 1:1:15
    B[i, 1] = 10
    B[i, 2] = -10
    B[i, 3] = 10
end
B
```

Out[13]:
```
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
```

7. Создайте 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, создайте следующие матрицы размерности 6 × 6.

In [14]:
```
z = zeros(Int64, 6, 6)
```

Out[14]:
```
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
```

In [15]:
```
E = ones(Int64, 6, 6)
```

Out[15]:
```
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
```

In [16]:
```
z1 = zeros(Int64, 6, 6)
for i in 1:1:6
    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
```

Out[16]:
```
6×6 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
```

In [ ]:
```
z1 = zeros(Int64, 6, 6)
for i in 1:1:6
    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 Array{Int64,2}:
 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
```

In [ ]:
```
z2 = zeros(Int64, 6, 6)
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 Array{Int64,2}:
 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
```

In [ ]:
```
z3 = zeros(Int64, 6, 6)
for i in 1:1:6
    z3[i,7-i] = 1
    if((7-i+2) <= 6) z3[i,9 - i] = E[i,9 - i] end
    if((7-i-2) >= 1) z3[i, 5 - i] = E[i, 5 - i] end
end
z3
```

```
6×6 Array{Int64,2}:
 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
```

In [ ]:
```
z4 = zeros(Int64, 6, 6)
for i in 1:1:6
    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 Array{Int64,2}:
 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
```

8. Напишите свою функцию, аналогичную функции outer() языка R. Функция должна иметь следующий интерфейс: outer(x,y,operation).

In [ ]:
```
function outer(x, y, operation)
    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
    c = zeros(size(x)[1], size(y)[2])
    for i in 1:size(x)[1], j in 1:size(y)[2], k in 1:size(x)[2]
        c[i, j] += operation(x[i, k], y[k, j])
    end
    return c
end
```

```
outer (generic function with 1 method)
```

In [ ]:
```
# проверим на данной матрице работу функции
X = [[1 1 3]; [5 2 6]; [-2 -1 -3]]
```

```
3×3 Array{Int64,2}:
  1   1   3
  5   2   6
 -2  -1  -3
```

In [ ]:
```
X * X
```

```
3×3 Array{Int64,2}:
  0   0   0
  3   3   9
 -1  -1  -3
```

In [ ]:
```
outer(X, X, *)
```

```
3×3 Array{Float64,2}:
  0.0   0.0   0.0
  3.0   3.0   9.0
 -1.0  -1.0  -3.0
```

In [ ]:
```
# Построим первую матрицу
A1 = outer(collect(0:4), collect(0:4)', +)
```

```
5×5 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 [ ]:
```
# построим вторую матрицу
A2 = outer(collect(0:4), collect(1:5)', ^)
```

```
5×5 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 [ ]:
```
# построим третью матрицу
# возьмем остаток от деления на 5
A3 = outer(collect(0:4), collect(0:4)', +).%5
```

```
5×5 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 [ ]:
```
# построим четвертую матрицу
# возьмем остаток от деления на 10
A4 = outer(collect(0:9), collect(0: 9)', +).%10
```

```
10×10 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 [ ]:
```
# построим пятую матрицу
# возьмем остаток от деления на 9
A5 = outer(collect(0:8), collect(-9:-1)', -).%9
```

```
9×9 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
```

9. Решите следующую систему линейных уравнений с 5 неизвестными.

In [ ]:
```
# зададим коэффициенты из системы уравнений
array_system = Array{Int64, 2}(undef, 5, 5)
m = 5
n = 5

for i in 1:1:m
    for j in 1:1:n
        array_system[i, j] = 1 + abs(i - j)
    end
end
println(round.(Int32, array_system))


# обозначим ответы
answers = [7; -1; -3; 5; 17]
```

```
Int32[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]
5-element Array{Int64,1}:
  7
 -1
 -3
  5
 17
```

In [ ]:
```
# с помощью обратной матрицы получим решение данной матрицы
array_system_inv = inv(array_system)
round.(Int, array_system_inv)
x_n = round.(Int, array_system_inv * answers)
```

```
5-element Array{Int64,1}:
 -2
  3
  5
  2
 -4
```

10. Создайте матрицу M размерности 6 × 10, элементами которой являются целые числа, выбранные случайным образом с повторениями из совокупности 1, 2, ... , 10.

In [ ]:
```
M = rand(1:10, 6, 10)
```

```
6×10 Array{Int64,2}:
  6  10   1   5   8   6   4  8   6   6
  8   8   7   2   9  10   5  3   7   4
 10   5   5   8   4   5  10  6   9   2
  3   1   9   6  10   6  10  1   6   9
  5   4   6   7   4   9   8  1   6   7
  6   7   5  10   8   7  10  4  10  10
```

- Найдите число элементов в каждой строке матрицы M, которые больше числа N (например, N = 4).

In [ ]:
```
N = 6
for i in 1:1:6
    count = 0
    for j in 1:1:10
        if M[i, j] > N
            count += 1
        end
    end
    println("Количество элементов больше ", N, " в строке = ", i, " равно ",
end
```

```
Количество элементов больше 6 в строке = 1 равно 3
Количество элементов больше 6 в строке = 2 равно 6
Количество элементов больше 6 в строке = 3 равно 4
Количество элементов больше 6 в строке = 4 равно 4
Количество элементов больше 6 в строке = 5 равно 4
Количество элементов больше 6 в строке = 6 равно 7
```

- Определите, в каких строках матрицы M число M (например, M = 7) встречается ровно 2 раза?

In [ ]:
```
M_ = 7

for i in 1:1:6
    count = 0
    for j in 1:1:10
        if M[i, j] == M_
            count += 1
        end
    end
    if count == 2
        println("Число ", M_, " встречается в строке ", i, " ровно 2 раза")
    end
end
```

```
Число 7 встречается в строке 2 ровно 2 раза
Число 7 встречается в строке 5 ровно 2 раза
Число 7 встречается в строке 6 ровно 2 раза
```

- Определите все пары столбцов матрицы M, сумма элементов которых больше K (например, K = 75).

In [ ]:
```
function sum_75(matrix, K)
    matrix_new = rand(10)
    for i in 1:1:10
        sum = 0
        for j in 1:1:6
            sum += matrix[j, i]
        end
        matrix_new[i] = sum
    end
    # теперь просуммируем все значения matrix_new попарно
    for i in 1:1:10
        for j in i+1:1:10
            sum = matrix_new[i] + matrix_new[j]
            if sum > K
                println("Столбцы - ", i, " и ", j)
            end
        end
    end
end
```

```
sum_75 (generic function with 1 method)
```

In [ ]:
```
sum_75(M, 75)
```

```
Столбцы - 1 и 4
Столбцы - 1 и 5
Столбцы - 1 и 6
Столбцы - 1 и 7
Столбцы - 1 и 9
Столбцы - 1 и 10
Столбцы - 2 и 5
Столбцы - 2 и 6
Столбцы - 2 и 7
Столбцы - 2 и 9
Столбцы - 3 и 5
Столбцы - 3 и 6
Столбцы - 3 и 7
Столбцы - 3 и 9
Столбцы - 4 и 5
Столбцы - 4 и 6
Столбцы - 4 и 7
Столбцы - 4 и 9
Столбцы - 4 и 10
Столбцы - 5 и 6
Столбцы - 5 и 7
Столбцы - 5 и 9
Столбцы - 5 и 10
Столбцы - 6 и 7
Столбцы - 6 и 9
Столбцы - 6 и 10
Столбцы - 7 и 9
Столбцы - 7 и 10
Столбцы - 9 и 10
```

11. Вычислите:

$$\sum_{i=1}^{20} \sum_{j=1}^{5} \frac{i^4}{3+j}$$

$$\sum_{i=1}^{20} \sum_{j=1}^{5} \frac{i^4}{3+ij}$$

In [ ]:
```
first_sum = 0
for i in 1:1:20
    for j in 1:1:5
        first_sum += (i^4)/(3+j)
    end
end
println(first_sum)
```

```
639215.2833333334
```

In [ ]:
```
second_sum = 0
for i in 1:1:20
    for j in 1:1:5
        second_sum += (i^4)/(3+i*j)
    end
end
println(second_sum)
```

```
89912.02146097136
```