

Лабораторная работа № 3

Покрас Илья Михайлович

2023, Москва

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

Ход работы

```

print("for cycle")
for i in 1:100
    print(i, " ", i^2, " ", i^3, " ")
end

println()
println("while cycle")
i = 1
while i <= 100
    print(i, " ", i^2, " ", i^3, " ")
    i = i + 1
end

squares = Dict{Int, Int}()
for i = 1:100
    squares = Dict{Int, Int}()
    i = i + 1
end

while i <= 100
    squares[i] = i^2
    i = i + 1
end

println("squares dict")
println(squares)

squares_arr = Dict{Int, Int}()
squares_arr = []
i = 1
while i <= 100
    push!(squares_arr, i^2)
    i = i + 1
end

println("squares_arr")
println(squares_arr)

println()
println("for cycle")
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 + 54 + 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 + 101 + 102 + 103 + 104 + 105 + 106 + 107 + 108 + 109 + 110 + 111 + 112 + 113 + 114 + 115 + 116 + 117 + 118 + 119 + 120 + 121 + 122 + 123 + 124 + 125 + 126 + 127 + 128 + 129 + 130 + 131 + 132 + 133 + 134 + 135 + 136 + 137 + 138 + 139 + 140 + 141 + 142 + 143 + 144 + 145 + 146 + 147 + 148 + 149 + 150 + 151 + 152 + 153 + 154 + 155 + 156 + 157 + 158 + 159 + 160 + 161 + 162 + 163 + 164 + 165 + 166 + 167 + 168 + 169 + 170 + 171 + 172 + 173 + 174 + 175 + 176 + 177 + 178 + 179 + 180 + 181 + 182 + 183 + 184 + 185 + 186 + 187 + 188 + 189 + 190 + 191 + 192 + 193 + 194 + 195 + 196 + 197 + 198 + 199 + 200 + 201 + 202 + 203 + 204 + 205 + 206 + 207 + 208 + 209 + 210 + 211 + 212 + 213 + 214 + 215 + 216 + 217 + 218 + 219 + 220 + 221 + 222 + 223 + 224 + 225 + 226 + 227 + 228 + 229 + 230 + 231 + 232 + 233 + 234 + 235 + 236 + 237 + 238 + 239 + 240 + 241 + 242 + 243 + 244 + 245 + 246 + 247 + 248 + 249 + 250 + 251 + 252 + 253 + 254 + 255 + 256 + 257 + 258 + 259 + 260 + 261 + 262 + 263 + 264 + 265 + 266 + 267 + 268 + 269 + 270 + 271 + 272 + 273 + 274 + 275 + 276 + 277 + 278 + 279 + 280 + 281 + 282 + 283 + 284 + 285 + 286 + 287 + 288 + 289 + 290 + 291 + 292 + 293 + 294 + 295 + 296 + 297 + 298 + 299 + 300 + 301 + 302 + 303 + 304 + 305 + 306 + 307 + 308 + 309 + 310 + 311 + 312 + 313 + 314 + 315 + 316 + 317 + 318 + 319 + 320 + 321 + 322 + 323 + 324 + 325 + 326 + 327 + 328 + 329 + 330 + 331 + 332 + 333 + 334 + 335 + 336 + 337 + 338 + 339 + 340 + 341 + 342 + 343 + 344 + 345 + 346 + 347 + 348 + 349 + 350 + 351 + 352 + 353 + 354 + 355 + 356 + 357 + 358 + 359 + 360 + 361 + 362 + 363 + 364 + 365 + 366 + 367 + 368 + 369 + 370 + 371 + 372 + 373 + 374 + 375 + 376 + 377 + 378 + 379 + 380 + 381 + 382 + 383 + 384 + 385 + 386 + 387 + 388 + 389 + 390 + 391 + 392 + 393 + 394 + 395 + 396 + 397 + 398 + 399 + 400 + 401 + 402 + 403 + 404 + 405 + 406 + 407 + 408 + 409 + 410 + 411 + 412 + 413 + 414 + 415 + 416 + 417 + 418 + 419 + 420 + 421 + 422 + 423 + 424 + 425 + 426 + 427 + 428 + 429 + 430 + 431 + 432 + 433 + 434 + 435 + 436 + 437 + 438 + 439 + 440 + 441 + 442 + 443 + 444 + 445 + 446 + 447 + 448 + 449 + 450 + 451 + 452 + 453 + 454 + 455 + 456 + 457 + 458 + 459 + 460 + 461 + 462 + 463 + 464 + 465 + 466 + 467 + 468 + 469 + 470 + 471 + 472 + 473 + 474 + 475 + 476 + 477 + 478 + 479 + 480 + 481 + 482 + 483 + 484 + 485 + 486 + 487 + 488 + 489 + 490 + 491 + 492 + 493 + 494 + 495 + 496 + 497 + 498 + 499 + 500 + 501 + 502 + 503 + 504 + 505 + 506 + 507 + 508 + 509 + 510 + 511 + 512 + 513 + 514 + 515 + 516 + 517 + 518 + 519 + 520 + 521 + 522 + 523 + 524 + 525 + 526 + 527 + 528 + 529 + 530 + 531 + 532 + 533 + 534 + 535 + 536 + 537 + 538 + 539 + 540 + 541 + 542 + 543 + 544 + 545 + 546 + 547 + 548 + 549 + 550 + 551 + 552 + 553 + 554 + 555 + 556 + 557 + 558 + 559 + 560 + 561 + 562 + 563 + 564 + 565 + 566 + 567 + 568 + 569 + 570 + 571 + 572 + 573 + 574 + 575 + 576 + 577 + 578 + 579 + 580 + 581 + 582 + 583 + 584 + 585 + 586 + 587 + 588 + 589 + 590 + 591 + 592 + 593 + 594 + 595 + 596 + 597 + 598 + 599 + 600 + 601 + 602 + 603 + 604 + 605 + 606 + 607 + 608 + 609 + 610 + 611 + 612 + 613 + 614 + 615 + 616 + 617 + 618 + 619 + 620 + 621 + 622 + 623 + 624 + 625 + 626 + 627 + 628 + 629 + 630 + 631 + 632 + 633 + 634 + 635 + 636 + 637 + 638 + 639 + 640 + 641 + 642 + 643 + 644 + 645 + 646 + 647 + 648 + 649 + 650 + 651 + 652 + 653 + 654 + 655 + 656 + 657 + 658 + 659 + 660 + 661 + 662 + 663 + 664 + 665 + 666 + 667 + 668 + 669 + 670 + 671 + 672 + 673 + 674 + 675 + 676 + 677 + 678 + 679 + 680 + 681 + 682 + 683 + 684 + 685 + 686 + 687 + 688 + 689 + 690 + 691 + 692 + 693 + 694 + 695 + 696 + 697 + 698 + 699 + 700 + 701 + 702 + 703 + 704 + 705 + 706 + 707 + 708 + 709 + 710 + 711 + 712 + 713 + 714 + 715 + 716 + 717 + 718 + 719 + 720 + 721 + 722 + 723 + 724 + 725 + 726 + 727 + 728 + 729 + 730 + 731 + 732 + 733 + 734 + 735 + 736 + 737 + 738 + 739 + 740 + 741 + 742 + 743 + 744 + 745 + 746 + 747 + 748 + 749 + 750 + 751 + 752 + 753 + 754 + 755 + 756 + 757 + 758 + 759 + 760 + 761 + 762 + 763 + 764 + 765 + 766 + 767 + 768 + 769 + 770 + 771 + 772 + 773 + 774 + 775 + 776 + 777 + 778 + 779 + 780 + 781 + 782 + 783 + 784 + 785 + 786 + 787 + 788 + 789 + 790
```

Рис. 1: Код выполнения пункта 1

```
a = parse(Int, readline())  
if isodd(a)  
    println("Нечётное")  
else  
    println(a)  
end
```

```
stdin> 3  
Нечётное
```

Рис. 2: Код выполнения пункта 2

```
function add_one(numb)
    println(numb + 1)
end

for i in 1:3
    a = parse{Int, readline()}
    add_one(a)
end
```

```
stdin> 2
3
stdin> 1
2
stdin> 4
5
```

```
println("Matrix1:")  
matrix1 = map(x -> x, reshape(Array(1:20), 4, 5))  
display(matrix1)  
  
println("\nMatrix2:")  
matrix2 = broadcast(x -> x, reshape(Array(1:20), 4, 5))  
display(matrix2)
```

```
Matrix1:  
4×5 Matrix{Int64}:  
 1  5  9 13 17  
 2  6 10 14 18  
 3  7 11 15 19  
 4  8 12 16 20
```

```
Matrix2:  
4×5 Matrix{Int64}:  
 1  5  9 13 17  
 2  6 10 14 18  
 3  7 11 15 19  
 4  8 12 16 20
```

Рис. 4: Код выполнения пункта 4

```
A = [1 1 3; 5 2 6; -2 -1 -3]
println("Matrix:")
display(A)

println("\nMatrix^3:")
display(A.^3)

for i in 1:3
    A[i, 3] = A[i, 1] + A[i, 2]
end

println("\nMatrix with sum:")
display(A)

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

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

Matrix with sum:
3×3 Matrix{Int64}:
 1  1  2
 5  2  7
-2 -1 -3
```

Рис. 5: Код выполнения пункта 5

```
B = zeros(15, 3)

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

Bt = transpose(B)
C = Bt * B

println("\nMultiplication matrix:")
display(C)
```

```
Multiplication matrix:
3×3 Matrix{Float64}:
 1500.0  -1500.0   1500.0
-1500.0   1500.0  -1500.0
 1500.0  -1500.0   1500.0
```

Рис. 6: Код выполнения пункта 6


```
NOTES = [[P 1 0 0 11, -1] 1 4344 0 and Apr 3 30 30] Apr 1 30 30)
breakpoint, ...
getopts("cd")
display(notepad)

NOTES = [[P 1 0 0 11, 0, -1] 1 4344 0 and Apr 3 30 30] Apr 1 30 30)
breakpoint, ...
getopts("cd")
display(notepad)

NOTES = [[P 1 0 0 11, 7, 7] 1 4344 0 and Apr 3 30 30] Apr 2 30 30)
breakpoint, ...
getopts("cd")
display(notepad)

NOTES = [[P 1 0 0 11, 1, 0, -1] 1 4344 0 and Apr 3 30 30] Apr 1 30 30)
breakpoint, ...
getopts("cd")
display(notepad)

20
w = ColumnVector{Vector{Int64}}()
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]

21
w = ColumnVector{Vector{Int64}}()
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]

22
w = ColumnVector{Vector{Int64}}()
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]

23
w = ColumnVector{Vector{Int64}}()
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]

24
w = ColumnVector{Vector{Int64}}()
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 0, 0]
```

Рис. 7: Код выполнения пункта 7

```
outer(x, y, operation) = transpose(hcat([[sum(operation(x[i, k], y[k, j]) for k in 1:size(x)[2]) for j in 1:size(y)[2]) for i in 1:size(x)[1]]...))

B = reshape(rand(1:5, 12), 2, 6)
A = reshape(rand(1:5, 12), 6, 2)
println("Source matrices:")
display(A)
display(B)
println("\nSum matrix:")
display(outer(A, B, +))

println("\nSub matrix:")
display(outer(A, B, -))

println("\nMul matrix:")
display(outer(A, B, *))

println("\nDiv matrix:")
display(outer(A, B, /))

println("\nPow matrix:")
display(outer(A, B, ^))

println("\nA1:")
display(outer(reshape(0:4, 5, 1), reshape(0:4, 1, 5), +))

println("\nA2:")
println(outer(reshape(0:4, 5, 1), reshape(1:5, 1, 5), ^))

println("\nA3:")
display(outer(hcat([[if i==j 1 else 0 end for j in 0:4] for i in 0:4]...), hcat([Vector(1:i+4).%5 for i in 0:4]...), *))

println("\nA4:")
display(outer(hcat([[if i==j 1 else 0 end for j in 0:9] for i in 0:9]...), hcat([Vector(1:i+9).%10 for i in 0:9]...), *))

println("\nA5:")
display(outer(hcat([[if i==j 1 else 0 end for j in 0:8] for i in 0:8]...), hcat([Vector(i+9:-1:i+1).%9 for i in 0:8]...), *))
```

Рис. 8: Код выполнения пункта 8

```
M = [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]
N = [7, -1, -3, 5, 17]
println("Solution Vector: ")
display(M \ N)
```

```
Solution Vector:
5-element Vector{Float64}:
-2.00000000000000036
 3.00000000000000058
 4.9999999999999998
 1.9999999999999991
-3.9999999999999999
```

Рис. 9: Код выполнения пункта 9

```
matrix = randi(10, 6, 10)
println("Matrix: ")
display(matrix)
num = 5

ans_vec1 = []
for i in 1:size(matrix)[1]
    counter = 0
    for j in 1:size(matrix)[2]
        if matrix[i, j] > num
            counter = counter + 1
        end
    end
    push!(ans_vec1, counter)
end
println("The amount of numbers, greater than N: ")
println(ans_vec1)

ans_vec2 = []
M = 7
for i in 1:size(matrix)[1]
    counter = 0
    for j in 1:size(matrix)[2]
        if matrix[i, j] == M
            counter = counter + 1
        end
    end
    push!(ans_vec2, counter)
end
println("The amount of numbers, that equal M: ")
println(ans_vec2)

K = 75
ans_vec3 = []
for i in 1:size(matrix)[2]-1 for j in 1:size(matrix)[2] if sum(matrix[i, i] + matrix[i, j]) > K
    println("The amount of column pairs, sum of which is greater than K: ")
    println(ans_vec3)
end

Matrix:
6x10 Matrix{Int64}:
9  2  5  2 10  6  6  5  8  6
4  5  3  4  7  9  9  5  3  4
6 10  4  4 10  6  5  4  1  8
4  5  3  7  6 10  9  9 10  3
7  3  2  3  2 10  5  1  4  9
5  5  6  2  2  2  2  1  6

The amount of numbers, greater than N:
Any{6, 4, 5, 6, 3, 2}

The amount of numbers, that equal M:
Any{0, 1, 0, 1, 1, 0}

The amount of column pairs, sum of which is greater than K:
[(1, 6), (5, 6), (6, 7), (6, 10)]
```

Рис. 10: Код выполнения пункта 10

```
A = [[i^4/(3+j) for j in 1:5] for i in 1:20]  
B = [[i^4/(3+i*j) for j in 1:5] for i in 1:20]  
display(sum(sum(A)))  
display(sum(sum(B)))
```

639215.2833333333

89912.02146097136

Рис. 11: Код выполнения пункта 11

Я освоил применение циклов функций и сторонних для Julia пакетов для решения задач линейной алгебры и работы с матрицами.