

Лабораторная работа №3

Управляющие структуры

Легиньких Г.А.

Российский университет дружбы народов, Москва, Россия

Информация

- Легиньких Галина Андреевна
- НФИбд-02-21
- Российский университет дружбы народов
- 1032216447@pfur.ru
- <https://github.com/galeginkikh>

Основная информация

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

1. Используя Jupyter Lab, повторите примеры из раздела 2.2.
2. Выполните задания для самостоятельной работы (раздел 2.4).

Выполнение

Циклы while и for

Для начала я изучила видеоматериал к лекции и повторила примеры из раздела 3.2. Сначала это были циклы while и for. Для различных операций, связанных с перебором индексируемых элементов структур данных, традиционно используются циклы while и for.

```
[2]: n = 0
while n < 10
    n += 1
    println(n)
end

1
2
3
4
5
6
7
8
9
10

[4]: 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!
```

Рис. 1: while


```
[8]: for n in 1:2:10
      println(n)
    end

1
3
5
7
9

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

[12]: # инициализация массива m x n из нулей:
      m, n = 5, 5
      A = fill{0, (m, n)}
```

Рис. 2: for вналогично while

Довольно часто при решении задач требуется проверить выполнение тех или иных условий. Для этого используют условные выражения. Повторила синтаксис условных выражений с тернарными операторами.

```
[24]: # используем `&&` для реализации операции "AND"  
# операция `%` вычисляет остаток от деления  
N = 20  
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
```

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

```
[26]: 10
```

Рис. 3: Условные выражения

Далее перешла к функциям. Julia дает нам несколько разных способов написать функцию. Первый требует ключевых слов `function` и `end`.

```
[28]: function sayhi(name)
      println("Hi $name, it's great to see you!")
      end
      sayhi("C-3PO")

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

[30]: # функция возведения в квадрат:
      function f(x)
          x^2
      end
      f(42)

[30]: 1764
```

Рис. 4: Функции 1-ый способ

В качестве альтернативы, можно объявить любую из выше определённых функций в одной строке.

```
[44]: sayhi2(name) = println("Hi $name, it's great to see you!")  
      sayhi2("C-3PO")  
  
      Hi C-3PO, it's great to see you!  
  
[46]: f2(x) = x^2  
      f2(42)  
  
[46]: 1764
```

Рис. 5: Функции 2-ой способ

Наконец, можно объявить выше определённые функции как “анонимные”.

```
[48]: sayhi3 = name -> println("Hi $name, it's great to see you!")  
      sayhi3("C-3P0")
```

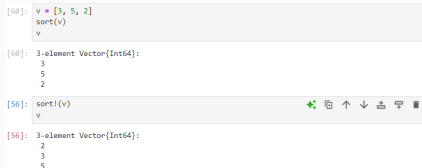
```
Hi C-3P0, it's great to see you!
```

```
[50]: f3 = x -> x^2  
      f3(42)
```

```
[50]: 1764
```

Рис. 6: Функции 3-ий способ

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



The screenshot shows a Julia REPL session with two examples. In the first, a variable `v` is assigned the value `[3, 5, 2]`, then `sort(v)` is called, and `v` is printed, showing it remains `[3, 5, 2]`. In the second, `sort!(v)` is called on the same variable, and `v` is printed, showing it has been modified to `[2, 3, 5]`.

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

[60]: 3-element Vector{Int64}:
      3
      5
      2

[56]: sort!(v)
      v

[56]: 3-element Vector{Int64}:
      2
      3
      5
```

Рис. 7: sort as !sort

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.

```
[62]: map(f, [1, 2, 3])  
  
[62]: 3-element Vector{Int64}:  
      1  
      4  
      9  
  
[64]: map(x -> x^2, [1, 2, 3])  
  
[64]: 3-element Vector{Int64}:  
      1  
      4  
      9
```

Рис. 8: `map`

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.

```

[250] %initMat([1, 2, 3])
[260] %reference Vector(1000);
      4
      4
[270] % Subtle warning: A
      A = 1 2 3 4 5 6 7 8 9 10
[280] %in VecMat(1000);
      1 2 3
      4 5 6
      7 8 9
[290] %END
[300] %in VecMat(1000);
      10 20 30
      40 50 60
      70 80 90
[310] % A = 1 2 3 4 5
[320] %in VecMat(1000);
      10 20 30
      40 50 60
[330] % A = 2 3 4 5 6 7 8 9 10
[340] %in VecMat(1000);
      10 20 30 40 50
      60 70 80 90 100
[350] %initMat([1] + 2 * 7 * 10) / 4
[360] %in VecMat(1000);
      10 20 30 40 50
      60 70 80 90 100
[370] %initMat([1] + 2 * 7 * 10) / 4, 10
[380] %in VecMat(1000);
      10 20 30 40 50
      60 70 80 90 100

```

Рис. 9: broadcast

Julia имеет более 2000 зарегистрированных пакетов, что делает их огромной частью экосистемы Julia. Научилась загружать пакеты.

```
[79]: import Pkg

[80]: Pkg.add("Example")

Resolving package versions...
No Changes to 'C:\Users\galin\.julia\environments\v1.10\Project.toml'
No Changes to 'C:\Users\galin\.julia\environments\v1.10\Manifest.toml'

[85]: Pkg.add("Colors")

Resolving package versions...
Installed FixedPointNumbers v0.8.5
Installed Reexport v1.2.2
Installed Colors v0.13.0
Installed ColorTypes v0.12.0
Updating 'C:\Users\galin\.julia\environments\v1.10\Project.toml'
[5ae59895] + Colors v0.13.0
Updating 'C:\Users\galin\.julia\environments\v1.10\Manifest.toml'
[3d0002f7] + ColorTypes v0.12.0
[5ae59895] + Colors v0.13.0
[93c48c17] + FixedPointNumbers v0.8.5
[189a3867] + Reexport v1.2.2
[37e2e46d] + LinearAlgebra
[2f01284e] + SparseArrays v1.10.0
[10745b16] + Statistics v1.10.0
[e66e0078] + CompilerSupportLibraries_jll v1.1.1+0
[4536629e] + OpenBLAS_jll v0.3.23+4
[hex8744a] + SuiteSparse_4ll v2.2.1+1
```

Рис. 10: Загрузка пакетов

Задание для самостоятельной работы (while)

- Задание 1.1 с помощью while

```
[97]: # Задание 1.1
      println("Числа и их квадраты:")
      i = 1
      while i <= 100
        println("Число: $i, Квадрат: $(i^2)")
        i += 1
      end
```

Числа и их квадраты:
Число: 1, Квадрат: 1
Число: 2, Квадрат: 4
Число: 3, Квадрат: 9
Число: 4, Квадрат: 16
Число: 5, Квадрат: 25
Число: 6, Квадрат: 36
Число: 7, Квадрат: 49
Число: 8, Квадрат: 64
Число: 9, Квадрат: 81
Число: 10, Квадрат: 100
Число: 11, Квадрат: 121
Число: 12, Квадрат: 144
Число: 13, Квадрат: 169
Число: 14, Квадрат: 196
Число: 15, Квадрат: 225
Число: 16, Квадрат: 256
Число: 17, Квадрат: 289

Рис. 11: Задание 1.1 while

Задание для самостоятельной работы (for)

- Задание 1.1 с помощью for

```
[99]: for i in 1:100
      println("Число: $i, Квадрат: ${i^2}")
      end
```

Число: 1, Квадрат: 1
Число: 2, Квадрат: 4
Число: 3, Квадрат: 9
Число: 4, Квадрат: 16
Число: 5, Квадрат: 25
Число: 6, Квадрат: 36
Число: 7, Квадрат: 49
Число: 8, Квадрат: 64
Число: 9, Квадрат: 81
Число: 10, Квадрат: 100
Число: 11, Квадрат: 121
Число: 12, Квадрат: 144
Число: 13, Квадрат: 169
Число: 14, Квадрат: 196
Число: 15, Квадрат: 225
Число: 16, Квадрат: 256
Число: 17, Квадрат: 289
Число: 18, Квадрат: 324

Рис. 12: Задание 1.1 for

- Задание 2

```
[118]: # Задание 2.1
function check_number(num)
    if num%2==0
        println(num)
    else
        println("нечётное")
    end
end
check_number(2)
2

[120]: check_number(5)
нечётное

[122]: # Задание 2.2
function check_number_ternary(num)
    println(num%2==0 ? num : "нечётное")
end
check_number_ternary(2)
2

[124]: check_number_ternary(5)
нечётное
```

Рис. 13: Задание 2

- Задание 3

```
[128]: # Задание 3
      add_one(x) = x + 1
      add_one(2)

[128]: 3

[130]: add_one(3)

[130]: 4

[136]: add_one_2 = x -> x + 1
      add_one_2(2)

[136]: 3
```

Рис. 14: Задание 3

- Задание 4

```
[150]: # Задание 4
      #, n = 3, 3
      A = reshape(1:9, n, n)
      A = map(add_one, A)

[150]: 3x3 Matrix{Int64}:
       2  5  8
       3  6  9
       4  7 10

[156]: # broadcast()
      A .= A .+ 1

[156]: 3x3 Matrix{Int64}:
       5  8 11
       6  9 12
       7 10 13
```

Рис. 15: Задание 4

- Задание 7.1

```
[192]: Z1 = Z
      for i in 1:6
        if i > 1
          Z1[i,i-1] = 1
        end
        if i < 6
          Z1[i,i+1] = 1
        end
      end
      Z1
```

```
[192]: 6x6 Matrix{Float64}:
 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
```

Рис. 16: Задание 7.1

Вывод

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