

# Лабораторная работа No 1.

---

Шалыгин Г. Э.

09 сентября 2023

Российский университет дружбы народов, Москва, Россия

Основная цель работы — подготовить рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомиться с основами синтаксиса Julia.

## **Выполнение лабораторной работы**

---

Первый пример использует ф-цию `println` для вывода строки на экран и вывода из буфера.

```
In [2]: println("Hello world")
```

```
Hello world
```

```
In [4]: io = IOBuffer()  
println(io, "Hello world")  
String(take!(io))
```

```
Out[4]: "Hello world\n"
```

```
In [ ]:
```

Получение типа выражения функцией `typeof`, вывод диапазонов типов, приведение типов с помощью.

```
In [5]: typeof(sqrt(3))
```

```
Out[5]: Float64
```

```
In [6]: for T in [Int8, Int16, Int32, Int64, Int128, UInt8, UInt16, UInt32, UInt64, UInt128]
println("$(@pad(T,7)): [$(typemin(T)), $(typemax(T))]" )
end
```

```
Int8: [-128,127]
Int16: [-32768,32767]
Int32: [-2147483648,2147483647]
Int64: [-9223372036854775808,9223372036854775807]
Int128: [-170141183460469231731687303715884105728,170141183460469231731687303715884105727]
UInt8: [0,255]
UInt16: [0,65535]
UInt32: [0,4294967295]
UInt64: [0,18446744073709551615]
UInt128: [0,340282366920938463463374607431768211455]
```

```
In [9]: Char(100), Bool(1)
```

```
Out[9]: ('d', true)
```

```
In [10]: promote(Int8(1), Float16(4.5), Float32(4.1))
```

```
Out[10]: (1.0f0, 4.5f0, 4.1f0)
```

Работа с функциями.

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

       g(x) = x^2

       println(f(4), ' ', g(4))

16 16
```

**Figure 3:** Задание функции

## Работа с векторами и матрицами.

```
In [13]: a = [4 7 6] # вектор-строка
         b = [1, 2, 3] # вектор-столбец
         a[2], b[2] # вторые элементы векторов a и b

Out[13]: (7, 2)
```

```
In [14]: a = 1; b = 2; c = 3; d = 4 # присвоение значений
         Am = [a b; c d] # матрица 2 x 2
         Am[1,1], Am[1,2], Am[2,1], Am[2,2] # элементы матрицы

Out[14]: (1, 2, 3, 4)
```

```
In [15]: aa = [1 2]
         AA = [1 2; 3 4]
         aa*AA*aa'
```

```
Out[15]: 1x1 Matrix{Int64}:
         27
```

**Figure 4:** Работа с векторами и матрицами

## Задания для самостоятельной работы

Использование функции `read` для чтения строк из файлов разными способами и из буфера.

```
In [22]: write("test.txt", "Hello world!")
```

```
Out[22]: 12
```

```
In [23]: open(io->read(io, String), "test.txt")
```

```
Out[23]: "Hello world!"
```

```
In [25]: io = open("test.txt", "r")  
         read(io, String)
```

```
Out[25]: "Hello world!"
```

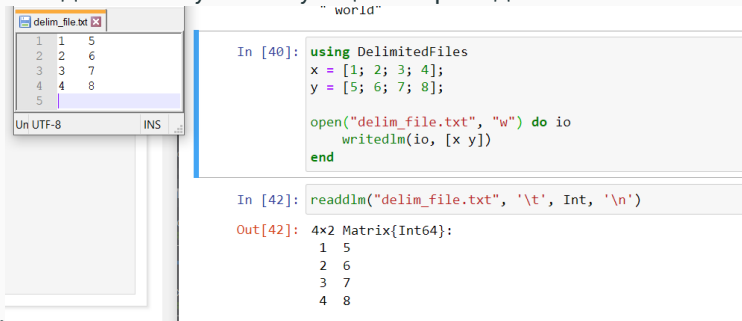
```
In [29]: io = IOBuffer("Hello world")  
         read(io, String)
```

```
Out[29]: "Hello world"
```



## Задания для самостоятельной работы

Использование функции `readdlm` для чтения значений из файла, разделенных символом. В данном случае табуляци и перевод



The screenshot displays a Jupyter Notebook interface. On the left, a file explorer window titled 'delim\_file.txt' shows a table with 5 rows and 3 columns of integers. The main notebook area contains two code cells. The first cell, labeled 'In [40]:', defines a `DelimitedFiles` object `io` and creates two 1x4 integer arrays `x` and `y`. It then opens 'delim\_file.txt' in write mode and writes the arrays `x` and `y` to the file using `writedlm`. The second cell, labeled 'In [42]:', uses `readdlm` to read the file back, specifying tab characters as delimiters and integers as the data type. The output, labeled 'Out[42]:', shows a 4x2 matrix of integers.

	1	2	3
1	1	5	
2	2	6	
3	3	7	
4	4	8	
5			

```
In [40]: using DelimitedFiles
x = [1; 2; 3; 4];
y = [5; 6; 7; 8];

open("delim_file.txt", "w") do io
    writedlm(io, [x y])
end

In [42]: readdlm("delim_file.txt", '\t', Int, '\n')

Out[42]: 4x2 Matrix{Int64}:
 1  5
 2  6
 3  7
 4  8
```

каретки.

## Задания для самостоятельной работы

Парсинг строки с приведением к значению определённого типа. Для целых значений указывается основание системы счисления.

```
In [51]: parse(Int, "1234"),  
         parse(Int, "1234", base = 5),  
         parse(Int, "afc", base = 16),  
         parse(Float64, "1.2e-3"),  
         parse(Complex{Float64}, "3.2e-1 + 4.5im"),  
         parse(Int, "100", base=2)
```

```
Out[51]: (1234, 194, 2812, 0.0012, 0.32 + 4.5im, 4)
```

**Figure 6:** parse

## Задания для самостоятельной работы

Арифметические и логические операции, можно выполнять с рациональными дробями.

```
In [52]: 2+2
```

```
Out[52]: 4
```

```
In [53]: 2*3
```

```
Out[53]: 6
```

```
In [54]: 3/4
```

```
Out[54]: 0.75
```

```
In [57]: 3//4 / 4//3
```

```
Out[57]: 9//16
```

```
In [58]: 16^0.5
```

```
Out[58]: 4.0
```

```
In [59]: sqrt(4) == 2
```

## Задания для самостоятельной работы

Алгебраические операции с векторами и матрицами, столбец и строка матрицы – векторы.

```
In [62]: v = [1 2 3]
         A = [0 0 1; 0 1 0; 1 0 0]
```

```
Out[62]: 3x3 Matrix{Int64}:
  0  0  1
  0  1  0
  1  0  0
```

```
In [71]: A[1:3,1] + v'
```

```
Out[71]: 3x1 Matrix{Int64}:
  1
  2
  4
```

```
In [73]: v - A[1, 1:3]'
```

```
Out[73]: 1x3 Matrix{Int64}:
  1  2  2
```

```
In [76]: v * A[1:3, 1] * 10
```

```
Out[76]: 1-element Vector{Int64}:
 30
```

В ходе выполнения работы подготовлено рабочее пространство и инструментарий для работы с языком программирования Julia, разобраны простейшие примеры синтаксиса Julia.