

# Лабораторная работа №1.

---

Евдокимов Иван Андреевич. НФИбд-01-20

10 ноября, 2023, Москва, Россия

Российский Университет Дружбы Народов

# Цель лабораторной работы

---

## Цель лабораторной работы

Подготовить рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомиться с основами синтаксиса Julia.

# **Процесс выполнения лабораторной работы**

---

## Пункт 1

1. Изучил документацию по основным функциям Julia для чтения / записи / вывода информации на экран: `read()`, `readline()`, `readlines()`, `readdlm()`, `print()`, `println()`, `show()`, `write()`. Привел свои примеры их использования, поясняя особенности их применения.

```
using DelimitedFiles
x = [1; 2; 3; 4];
y = [5; 6; 7; 8];
z = [5; 6; 7; 8];
open("delim_file.txt", "w") do io
    writedlm(io, [x y z])
end
```

Рис. 1: Подготовка 1

## 1.1. Вывод с помощью read():

```
read("delim_file.txt", String)
```

```
"1\t5\t5\n2\t6\t6\n3\t7\t7\n4\t8\t8\n"
```

Рис. 2: read()

## 1.2. Вывод с помощью readline():

```
readline("delim_file.txt")
```

```
"1\t5\t5"
```

**Рис. 3:** readline()

## 1.3. Вывод с помощью readlines():

```
readlines("delim_file.txt")
```

```
4-element Vector{String}:
```

```
"1\t5\t5"
```

```
"2\t6\t6"
```

```
"3\t7\t7"
```

```
"4\t8\t8"
```

**Рис. 4:** readlines()



## 1.4. Вывод с помощью readdlm():

```
readdlm("delim_file.txt", '\t', Int, '\n')
```

```
4x3 Matrix{Int64}:
```

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

**Рис. 5:** readdlm()

## 1.5. Запись с помощью write():

```
# Открываю файл для записи
a = [1; 2; 3; 4]
file = open("output.txt", "w")
# Строка для вывода
output_string = "Привет, мир!"
# Используем функцию write() для записи строки в файл
write(file, output_string)
# Закрываем файл
close(file)
```

Рис. 6: write()

## 1.6. Вывод с помощью print()

```
print(a, " ", output_string)  
print(a, " ", output_string)
```

[1, 2, 3, 4] Привет, мир! [1, 2, 3, 4] Привет, мир!

**Рис. 7:** print()

## 1.7. Вывод с помощью println():

```
println(a, " ", output_string)  
println(a, " ", output_string)
```

[1, 2, 3, 4] Привет, мир!

[1, 2, 3, 4] Привет, мир!

**Рис. 8:** println()

## 1.8. Вывод с помощью show():

```
show(a)  
show(output_string)
```

[1, 2, 3, 4] "Привет, мир!"

**Рис. 9:** show()

2. Изучите документацию по функции `parse()`. Приведите свои примеры её использования, поясняя особенности её применения.

```
parse(Int, "1234")
```

1234

```
parse(Int, "1234", base = 5)
```

194

```
parse(Int, "afc", base = 16)
```

2812

```
parse(Float64, "1.2e-3")
```

0.0012

```
parse(Complex{Float64}, "3.2e-1 + 4.5im")
```

0.32 + 4.5im

Рис. 10: `parse()`

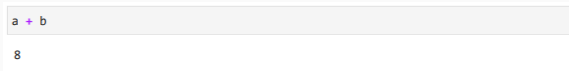
3. Изучите синтаксис Julia для базовых математических операций с разным типом переменных: сложение, вычитание, умножение, деление, возведение в степень, извлечение корня, сравнение, логические операции. Приведите свои примеры с пояснениями по особенностям их применения.

```
a,b,c,d = 5,3,true,false
```

```
(5, 3, true, false)
```

**Рис. 11:** Исходные файлы

### 3.1. Операция сложение:



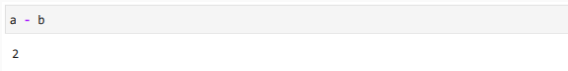
The diagram illustrates the addition operation. It consists of two horizontal rectangular boxes. The top box is light gray and contains the expression  $a + b$ , where the plus sign is purple. The bottom box is white and contains the number 8.

$$a + b$$
$$8$$

**Рис. 12:** Сложение



## 3.2. Операция вычитание:



The diagram illustrates the subtraction operation. It consists of two horizontal rectangular boxes. The top box is light gray and contains the expression  $a - b$ , where 'a' is black, '-' is purple, and 'b' is black. The bottom box is white and contains the number '2' in black.

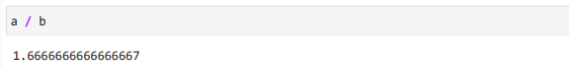
**Рис. 13:** Вычитание

### 3.3. Операция умножение:

$a * b$
15

**Рис. 14:** Умножение

### 3.4. Операция деление:



a / b

1.6666666666666667

**Рис. 15:** Деление

### 3.5. Операция возведение в степень:

$a^b$
125

**Рис. 16:** Возведение в степень

## 3.6. Операция извлечение корня:

```
sqrt(a)
```

```
2.23606797749979
```

**Рис. 17:** Извлечение корня

## 3.7. Операция сравнение:

```
a == b
```

```
false
```

**Рис. 18:** Сравнение

## 3.8. Логические операции:

```
-3 # Побитовое НЕ
-6
a & b # Побитовое И
1
a | b # Побитовое ИЛИ
7
a ^ b # Побитовое исключающее ИЛИ
syntax: colon expected in "?" expression
Stacktrace:
[1] top-level scope
      @ In[28]:1
a >>> b # Логический сдвиг вправо
0
```

Рис. 19: Логические операции 1

```
a >> b # Побитовый/логический сдвиг вправо
0
a << b # Побитовый/логический сдвиг влево
40
c && d # Логическое И
false
c || d # Логическое ИЛИ
true
!c # Логическое НЕ
false
```

Рис. 20: Логические операции 2

4. Приведите несколько своих примеров с пояснениями с операциями над матрицами и векторами: сложение, вычитание, скалярное произведение, транспонирование, умножение на скаляр.

```
A = [1 2; 2 3]  
B = [3 2; 2 3]  
C = [21; 45]
```

```
2-element Vector{Int64}:  
 21  
 45
```

**Рис. 21:** Подготовка 2



## 4.1. Сумма матриц:

A + B

2x2 Matrix{Int64}:

4 4

4 6

**Рис. 22:** Сумма матриц

## 4.2. Разность матриц:

A - B

2x2 Matrix{Int64}:

-2 0

0 0

**Рис. 23:** Разность матриц

## 4.3. Транспонирую матрицу A:

```
transpose(A)
```

```
2x2 transpose(::Matrix{Int64}) with eltype Int64:
```

```
1 2
```

```
2 3
```

**Рис. 24:** Транспонирую матрицу A

## 4.4. Умножение матрицы на число:

A \* 2

2x2 Matrix{Int64}:

2 4

4 6

**Рис. 25:** Умножение матрицы на число

## 4.5. Умножение матрицы на матрицу:

```
A * B  
2x2 Matrix{Int64}:  
 7  8  
12 13
```

**Рис. 26:** Умножение матрицы на матрицу

## **Выводы**

---

Подготовил рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомился с основами синтаксиса Julia.