

# Лабораторная работа №1

Julia. Установка и настройка. Основные принципы.

Герра Гарсия Паола Валентина

## Содержание

Цель работы .....	1
Задание .....	2
Теоретическое введение .....	2
Выполнение лабораторной работы .....	2
Выводы .....	8
Список литературы.....	8

## Список иллюстраций

Запуск Julia .....	2
Выполнение примеров из лабораторной .....	3
Выполнение примеров из лабораторной .....	3
Выполнение примеров из лабораторной .....	3
Чтение файла.....	4
Вывод на печать .....	5
Вывод на печать .....	5
Команда записи .....	6
Документация по функции parse() .....	6
Примеры использования функции parse() .....	6
Примеры базовых математических операций .....	7
Примеры базовых математических операций .....	7
примеры операций над матрицами.....	8
примеры операций над векторами.....	8

## Цель работы

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

## Задание

1. Установите под свою операционную систему Julia, Jupyter.
2. Используя Jupyter Lab, повторите примеры из раздела лабораторной работы.
3. Выполните задания для самостоятельной работы.

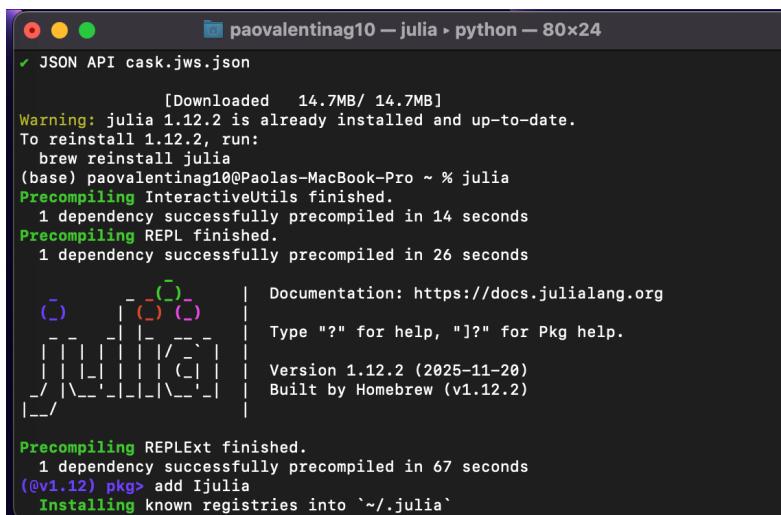
## Теоретическое введение

Julia – высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений [[@julialang](#)]. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков, однако имеет некоторые существенные отличия.

Для выполнения заданий была использована официальная документация Julia [[@juliadoc](#)].

## Выполнение лабораторной работы

У меня уже были установлены Julia и Jupyter (рис. [-@fig:001]).



```
paovalentinag10 — julia > python — 80x24
✓ JSON API cask.jws.json
[Downloaded 14.7MB/ 14.7MB]
Warning: julia 1.12.2 is already installed and up-to-date.
To reinstall 1.12.2, run:
brew reinstall julia
(base) paovalentinag10@Paolas-MacBook-Pro ~ % julia
Precompiling InteractiveUtils finished.
1 dependency successfully precompiled in 14 seconds
Precompiling REPL finished.
1 dependency successfully precompiled in 26 seconds

Documentation: https://docs.julialang.org
Type "?" for help, "]?" for Pkg help.
Version 1.12.2 (2025-11-20)
Built by Homebrew (v1.12.2)

Precompiling REPLExt finished.
1 dependency successfully precompiled in 67 seconds
(@v1.12) pkg> add IJulia
Installing known registries into `~/.julia`
```

### Запуск Julia

Теперь повторим простейшие примеры для знакомства с синтаксисом Julia (рис. [-@fig:001]-[-@fig:004]).

**Рис. 1.4. Пример получения информации о дате и пользователе OS Linux в Jupyter**

**1.3.3. Основы синтаксиса Julia на примерах**

Синтаксис синтаксис языка Julia очень похож на языки C (–S). Далее приведены простейшие примеры с использованием синтаксиса Julia, написанные в блокноте Jupyter Lab.

**Определение функции в листинге:**

```
function f(x)
    end
    f(1)
```

Здесь определена одна функция `f`, которая имеет один параметр `x`. Результатом выполнения выражения `f(3)` или `f(4)` будет значение числа `x` либо операции, например,  $3^2$  или  $\sqrt{3} - 4$ , значение чисто `x`.

Далее приведены примеры с использованием циклов `for` и `while` и отсутствием какого-либо значения. Такие выражения получают результат, равный единице на шаге, а также могут быть допущены частные выражения, не имеющие смысла.

Для определения кратных значений диапазона можно использовать числовые выражения:

```
for i = 1:10, Int64, Int32, Int64, Int128, Int64, Int64, Int64, Int64, Int64, Int64
```

В результате получим ненулевые и максимальные значения целочисленных типов.

В Julia преобразование типов можно реализовать или прямым указанием, например, явное преобразование числа 2.0 преобразовать в целое, а число 2 в список:

## Выполнение примеров из лабораторной

**Рис. 1.6. Примеры определения функций в одном типе**

function f(x)
 end
 f(1)

Для определения нескольких методов функции:  
«`generic`» + «`имя функции`» + «`типы`»

Например, для вектора строк:

```
g1(x::Vector{String})
```

Пример определения одномерных массивов (вектор-строка и вектор-столбец) и обращение к их вторым элементам:

```
a = [1 2] # вектор-строка
b = [1; 2] # вектор-столбец
c = [1, 2] # второй элемент вектора a и b
```

Пример определения для первого массива (матрицы) и обращение к его элементам:

```
function f(x)
    end
    f(1)
```

**Лаб1.3.2. Слайд. Компьютерная практика на языке Julia**

Функции f(x)  
end  
f (generic function with 1 method)  
f(x)  
x  
x[1]=2  
x (generic function with 1 method)  
x[1]

## Выполнение примеров из лабораторной

**Рис. 1.7. Примеры определения функций**

`a = 1; b = 2; c = 3; d = 4` – предварительный назначенный

`Ab = [a b; c d]` – вектор-строка, 2 × 2

`Ac = [a b; c d]; Ad = [c d; a b]` – элементы матрицы (вектор-строка, 2 × 2)

Пример выполнения операций над массивами («`aa'` – транспонирование вектора (матрицы), `aa * bb'` – умножение векторов, `aa' * bb` – умножение матриц).

`aa = [1 2]`  
`bb = [1; 2]`  
`aa' = 2`  
`aa * bb' = 3`  
`aa' * bb = 2`

`aa = [1 2; 3 4]`  
`bb = [1 2; 3 4]`  
`aa' = [1 3; 2 4]`  
`aa * bb' = 10`  
`aa' * bb = 10`

`Ab = [1 2; 3 4]`  
`Ac = [1 2; 3 4]`  
`Ad = [3 4; 1 2]`  
`Ab * Ac = 10`  
`Ac * Ad = 10`  
`Ab * Ad = 10`

`Ab = [1 2; 3 4]`  
`Ac = [1 2; 3 4]`  
`Ad = [3 4; 1 2]`  
`Ab * Ac = 10`  
`Ac * Ad = 10`  
`Ab * Ad = 10`

**Рис. 1.8. Примеры работы с массивами**

Лабораторная работа № 1. Julia. Установка и настройка. Основные принципы.

**1.3.4. Задания для самостоятельной работы**

## Выполнение примеров из лабораторной

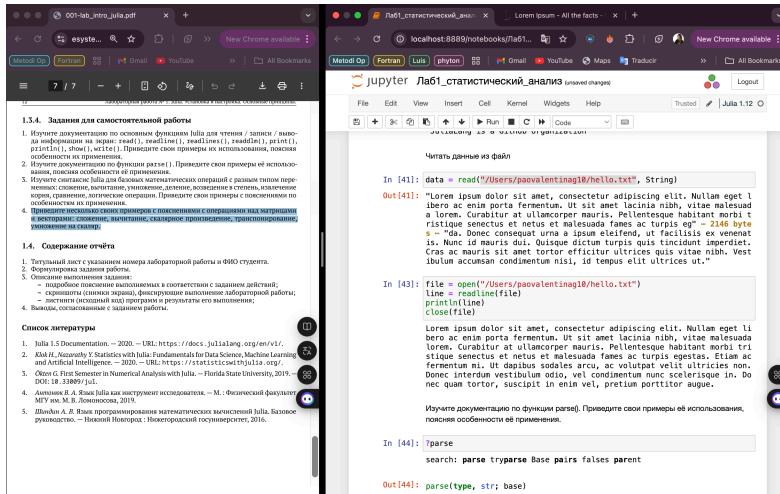
Теперь перейдем к выполнению заданий.

## Задание №1

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

Для того, чтобы ознакомиться с документацией достаточно поставить знак `? перед интересующей функцией. Например, ?print.`

Создадим текстовый файл с любым содержанием в папке, где мы работаем. Откроем его на чтение и прочитаем с помощью команды `read()`. Текст выведется в одну строку с разделителями `\r\n`. Также прочитаем текст используя функцию `readline()` - выведется только первая строка. Чтобы прочитать все строки в файле используем команду `readlines()` (рис. [-@fig:005]).



## Чтение файла

Далее посмотрим, как работает команда `println()`, `print()` и `show()` (рис. [-@fig:006]-[-@fig:007]).

Лаб1\_статистический\_анализ (untracked changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Logout

julia> io = IOBuffer("Julialang is a GitHub organization");

julia> read(io, String)

"Julialang is a GitHub organization"

Читать дальше из файла

In [41]: data = read("W:\Users\paiva\Downloads\HelloWorld.txt", String)

Out[41]: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam eget libero ac enim porta fermentum. Ut sit amet lacinia nibh, vitae malesuada a lorem. Curabitur at ultricies mauris. Pellentesque habitant morbi t risus senectus et netus et malesuada fames ac turpis egestas. 214 byte s. Sed id nunc id mauris dui. Quisque dictum turpis quis tincidunt imperdiet. Cras ac nulla sit amet tortor efficitur ultricies quis nibh. Vest ac condimentum condimentum nisi, id tempus tellus utriles ut.

In [43]: file = open("W:\Users\paiva\Downloads\HelloWorld.txt")

line = readline(file)

print(line)

close(file)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam eget libero ac enim porta fermentum. Ut sit amet lacinia nibh, vitae malesuada a lorem. Curabitur at ultricies mauris. Pellentesque habitant morbi t risus senectus et netus et malesuada fames ac turpis egestas. 214 byte s. Sed id nunc id mauris dui. Quisque dictum turpis quis tincidunt imperdiet. Cras ac nulla sit amet tortor efficitur ultricies quis nibh. Vest ac condimentum condimentum nisi, id tempus tellus utriles ut.

Изучите документацию по функции `readfile`. Приведите свои примеры её использования, включая особенности её применения.

## *Вывод на печать*

Лабораторная работа №1

Лаб 1. Статистический анализ

Файл: lab1\_intro\_julia.pdf

Браузер: Google Chrome

Логин: metod01@mail.ru

Пароль: Fortran1

Логин: lab1stat@yandex.ru

Пароль: New chrome available

Логин: metod01@mail.ru

Пароль: Fortran1

Логин: julia1.12

Пароль: Logout

Файл View Insert Cell Kernel Widgets Help Trusted Julia 1.12

;date

In [21]: ;whoami

paavolintanogl0

In [22]: typeof(3), typeof(3.5), typeof(3.55), typeof(sqrt(3+im)), typeof(pi)

Out[22]: (Int64, Float64, Float64, ComplexF64, Irrational{Int})

In [23]: 1.0+0.j, 1.0-(0,-0).j, 0.0+0.j

Out[23]: (Inf, -Inf, NaN)

In [25]: typeof(1.0+0.j), typeof(1.0-(0,-0.j)), typeof(0.0+0.j)

Out[25]: (Float64, Float64, Float64)

In [26]: for i in [Int8, Int16, Int32, Int64, Int128, UInt8, UInt16, UInt32, UInt64, UInt128]
 println("\$(typemin(i))..\$(typemax(i))")
end

[1]: [-128, 127]

[2]: [-32768, 32767]

[3]: [-2147483648, 2147483647]

[4]: [-9223372036854775808, 9223372036854775807]

[5]: [-18446744073709553615, 18446744073709553615]

[6]: [-34028236692093844337467341768211455]

Рис. 1.5. Примеры определения типа числовых величин

Лабораторная работа №1

Файл: lab1\_intro\_julia.pdf

Браузер: Google Chrome

Логин: metod01@mail.ru

Пароль: Fortran1

Логин: lab1stat@yandex.ru

Пароль: New chrome available

Логин: metod01@mail.ru

Пароль: Fortran1

Логин: julia1.12

Пароль: Logout

Файл View Insert Cell Kernel Widgets Help Trusted Julia 1.12

;date

In [21]: ;whoami

paavolintanogl0

In [22]: typeof(3), typeof(3.5), typeof(3.55), typeof(sqrt(3+im)), typeof(pi)

Out[22]: (Int64, Float64, Float64, ComplexF64, Irrational{Int})

In [23]: 1.0+0.j, 1.0-(0,-0).j, 0.0+0.j

Out[23]: (Inf, -Inf, NaN)

In [25]: typeof(1.0+0.j), typeof(1.0-(0,-0.j)), typeof(0.0+0.j)

Out[25]: (Float64, Float64, Float64)

In [26]: for i in [Int8, Int16, Int32, Int64, Int128, UInt8, UInt16, UInt32, UInt64, UInt128]
 println("\$(typemin(i))..\$(typemax(i))")
end

[1]: [-128, 127]

[2]: [-32768, 32767]

[3]: [-2147483648, 2147483647]

[4]: [-9223372036854775808, 9223372036854775807]

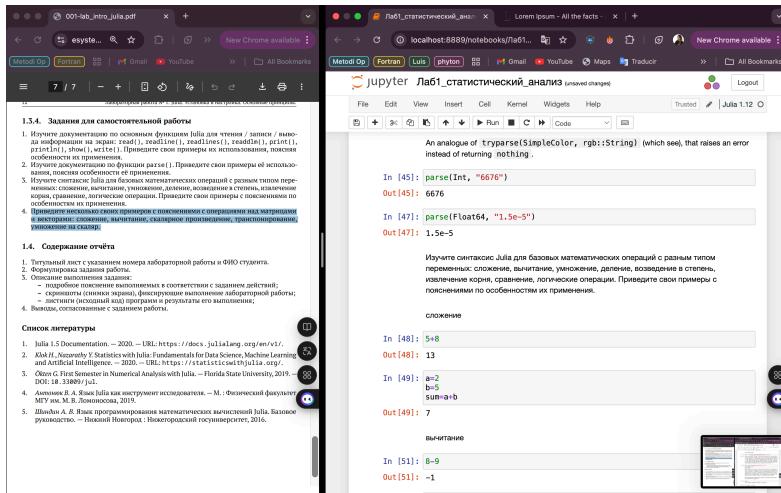
[5]: [-18446744073709553615, 18446744073709553615]

[6]: [-34028236692093844337467341768211455]

Рис. 1.6. Примеры приведения аргументов к единому типу

## *Вывод на печать*

Посмотрим на работу функции `write()` (рис. [-@fig:008])



## Команда записи

### Задание №2

Изучим документацию по функции `parse()` (рис. [-@fig:009]). Приведем свои примеры её использования, поясняя особенности её применения (рис. [-@fig:010]).

```
In [44]: ?parse
search: parse tryparse Base pairs falses parent

Out[44]: [begin(verbatim)
parse(type::str, base)
end(verbatim)]
```

Parse a string as a number. For `\texttt{Int}` types, a base can be specified (the default is 10). For floating-point types, the string is parsed as a decimal floating-point number. `\texttt{Complex}` types are parsed from decimal strings of the form `\texttt{"R}+im"` as a `\texttt{\text{Complex}(R,im)}` of the requested type; `\texttt{"r"}+im` or `\texttt{r}+im` can also be used instead of `\texttt{"R}+im"`, and `\texttt{"R"}+im` or `\texttt{R}+im` are also permitted. If the string does not contain a valid number, an error is raised.

```
[begin(quote)
textbf{compat}
Julia 1.1

\texttt{parse(parse(Bool, str))} requires at least Julia 1.1.
end(quote)]
```

`\section{Examples}`

```
[begin(verbatim)
julia> parse(Int, "1234")
1234

julia> parse(Int, "1234", base = 5)
194

julia> parse(Int, "afc", base = 16)
2812

julia> parse(Float64, "1.2e-3")
0.0012

julia> parse(Complex{Float64}, "3.2e-1 + 4.5im")
0.32 + 4.5im
end(verbatim)]
```

`\rule{width}{height}[1pt]`

## Документация по функции `parse()`

```
In [45]: parse(Int, "6676")
Out[45]: 6676

In [47]: parse(Float64, "1.5e-5")
Out[47]: 1.5e-5
```

## Примеры использования функции `parse()`

### Задание №3

Изучим синтаксис Julia для базовых математических операций с разным типом переменных: сложение, вычитание, умножение, деление, возведение в степень,

извлечение корня, сравнение, логические операции. Приведем примеры с пояснениями по особенностям их применения (рис. [-@fig:011]-[-@fig:012]).

The screenshot shows a Jupyter Notebook interface with the title "Jupyter Lab1\_статистический\_анализ". The notebook contains the following code cells:

- In [54]:  $5-6$   
Out [54]: -1
- In [55]:  $a=7$   
 $b=8$   
 $diff=a-b$   
Out [55]: -1
- умножение  
In [56]:  $a*b$   
Out [56]: 56
- деление  
In [57]:  $a/b$   
Out [57]: 0.875
- степень  
In [58]:  $a^b$   
Out [58]: 5764801
- In [59]:  $\sqrt{58}$   
Out [59]: 7.615773105863909
- In [60]:  $a=b$   
Out [60]: false

### Примеры базовых математических операций

The screenshot shows a Jupyter Notebook interface with the title "Jupyter Lab1\_статистический\_анализ". The notebook contains the following code cells:

- In [56]:  $a*b$   
Out [56]: 56
- деление  
In [57]:  $a/b$   
Out [57]: 0.875
- степень  
In [58]:  $a^b$   
Out [58]: 5764801
- In [59]:  $\sqrt{58}$   
Out [59]: 7.615773105863909
- In [60]:  $a=b$   
Out [60]: false

### Примеры базовых математических операций

## Задание №4

Приведем несколько примеров с пояснениями с операциями над матрицами и векторами: сложение, вычитание, скалярное произведение, транспонирование, умножение на скаляр (рис. [-@fig:013]-[-@fig:014]).

Jupyter Лаб1\_статистический\_анализ Last Checkpoint: el sábado pasado a las 23:01 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Julia 1.12

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

```
In [61]: A=[1 2; 3 4]
B=[6 7; 8 9]
A*B

Out[61]: 2x2 Matrix{Int64}:
 7  9
11 13

In [62]: A-B

Out[62]: 2x2 Matrix{Int64}:
 -5 -5
 -5 -5

In [63]: using LinearAlgebra
v1=[1,2,3]
v2=[4,5,6]
dot(v1,v2)

Out[63]: 32

In [64]: A'

Out[64]: 2x2 adjoint(::Matrix{Int64}) with eltype Int64:
 1  3
 2  4

In [65]: 2*A

Out[65]: 2x2 Matrix{Int64}:
 2  4
 6  8
```

### примеры операций над матрицами

```
In [63]: using LinearAlgebra
v1=[1,2,3]
v2=[4,5,6]
dot(v1,v2)

Out[63]: 32

In [64]: A'

Out[64]: 2x2 adjoint(::Matrix{Int64}) with eltype Int64:
 1  3
 2  4

In [65]: 2*A

Out[65]: 2x2 Matrix{Int64}:
 2  4
 6  8
```

### примеры операций над векторами

## Выводы

В результате выполнения данной лабораторной работы я подготовила рабочее пространство и инструментарий для работы с языком программирования Julia, на простейших примерах познакомилась с основами синтаксиса Julia.

## Список литературы