Лабораторная работа №1

Julia. Установка и настройка. Основные принципы.

Замбалова Дина Владимировна

Содержание

Список иллюстраций

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 [1]. 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 [2].

Выполнение лабораторной работы

Так как у меня уже были установлены Julia и Jupyter (рис. 1),

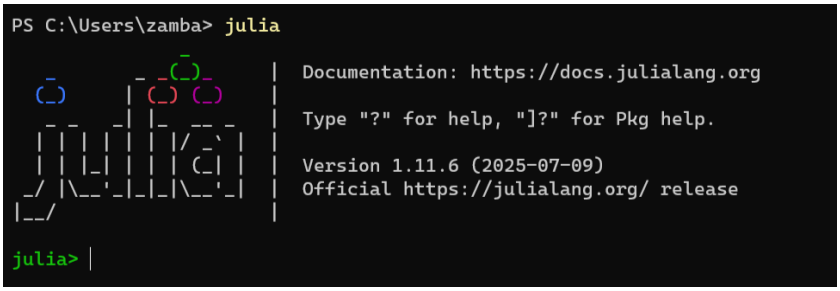

Рис. 1: Запуск Julia

повторим простейшие примеры для знакомства с синтаксисом Julia (рис. 2-4).

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

[4]: io = IOBuffer();

[5]: println(io, "Hello, world")

[6]: String(take!(io))

[6]: "Hello, world\n"
```

Рис. 2: Выполнение примеров из лабораторной

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

[24]: f (generic function with 1 method)

[25]: f(3)

[25]: 9

[26]: g(x)=x^2

[26]: g (generic function with 1 method)

[27]: g(9)

[27]: 81
```

Рис. 3: Выполнение примеров из лабораторной

```
[31]: a = 1; b = 2; c = 3; d = 4

[31]: 4

[32]: Am = [ a b; c d]

[32]: 2×2 Matrix{Int64}:
       1  2
       3  4

[33]: Am[1,1], Am[1,2], Am[2,1], Am[2,2]

[33]: (1, 2, 3, 4)

[34]: aa = [1 2]

[34]: 1×2 Matrix{Int64}:
       1  2

[35]: AA = [1 2; 3 4]

[35]: 2×2 Matrix{Int64}:
       1  2
       3  4

[36]: aa*AA*aa'

[36]: 1×1 Matrix{Int64}:
       27
```

Рис. 4: 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 вывелся 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. 5).

```
[51]: n = open("hello.txt", "r")

[51]: IOStream(<file hello.txt>)

[52]: data = read(n, String)

[52]: "Hello, world!\r\nBonjour, monde!\r\n¡Hola, mundo!\r\n"

[53]: m = open("hello.txt", "r")
      lines = readline(m)

[53]: "Hello, world!"

[54]: m = open("hello.txt", "r")
      lines = readlines(m)

[54]: 3-element Vector{String}:
       "Hello, world!"
       "Bonjour, monde!"
       "¡Hola, mundo!"
```

Рис. 5: Чтение файла

Далее посмотрим, как работает команда `println()`, `print()` и `show()` (рис. 6-7).

```
[57]: using DelimitedFiles
      x =[1; 2; 3; 4];
      y= ["a"; "b"; "c"; "d"];

      open("hello.txt", "w") do io
          for i in 1:length(x)
              println(io,string(x[i], " ", y[i]))
          end
      end

      readdlm("hello.txt")
```

```
[57]: 4×2 Matrix{Any}:
       1  "a"
       2  "b"
       3  "c"
       4  "d"
```

Рис. 6: Вывод на печать

```
[65]: for i in 1:1:3
          print(i, " ")
      end
```

```
1 2 3
```

```
[66]: for i in 1:1:3
          println(i, " ")
      end
```

```
1
2
3
```

```
[59]: show("Hello, world")
```

```
"Hello, world"
```

```
[60]: print("Hello, world")
```

```
Hello, world
```

Рис. 7: Вывод на печать

Посмотрим на работу функции `write()` (рис. 8).

```
[61]: io = IOBuffer()
      write(io, "Hello", ", world")

[61]: 12

[62]: String(take!(io))

[62]: "Hello, world"

[63]: write(io, "Hello") + write(io, ", world")

[63]: 12

[64]: String(take!(io))

[64]: "Hello, world"
```

Рис. 8: Команда записи

Задание №2

Изучим документацию по функции parse() (рис. 9). Приведем свои примеры её использования, поясняя особенности её применения (рис. 10).

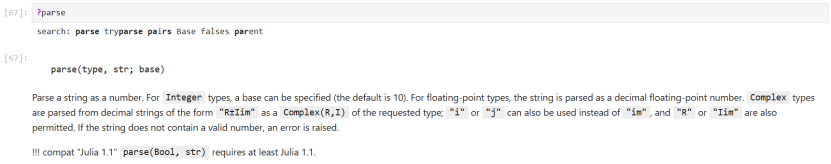

Рис. 9: Документация по функции parse()

```
[68]: parse(Int, "125647")

[68]: 125647

[69]: parse(Float64, "3.14")

[69]: 3.14

[70]: parse(Int, "14"; base = 5)

[70]: 9

[73]: parse(Int, "hello", base = 60)

[73]: 566092070
```

Рис. 10: 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. 11-12).

[79]:
```
a = 6 + 6
b = 3.14 + 41.3
c = 8.8 * 9.9
d = "Hell" * "o"
println(a, '\n', b, '\n', c, '\n', d)
```

```
12
44.44
87.12
Hello
```

[75]:
```
a = 6 - 6
b = 3.14 - 41.3
println(a, '\n', b)
```

```
0
-38.16
```

[76]:
```
a = 6 / 6
b = 3.14 / 41.3
println(a, '\n', b)
```

```
1.0
0.07602905569007265
```

[78]:
```
a = 6^6
b = 3^4
println(a, '\n', b)
```

```
46656
81
```

Рис. 11: Примеры базовых математических операций

```
[81]: a = sqrt(121)

[81]: 11.0

[83]: a = 3 > 5
      b = 2 == 2
      c = 10 < 11
      d = 5 != 5
      println(a, '\n', b, '\n', c, '\n', d)

      false
      true
      true
      false

[84]: a = true && true
      b = false && true
      println(a, '\n', b)

      true
      false

[85]: a = true || true
      b = false || true
      println(a, '\n', b)

      true
      true

[86]: a = !true
      b = !false
      println(a, '\n', b)

      false
      true
```

Рис. 12: 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. 13-14).

```
[87]: A = [4 9; 2 3]
      B = [1 1; 1 1]
      C = A + B
      C
```

```
[87]: 2×2 Matrix{Int64}:
       5  10
       3   4
```

```
[89]: A = [4 9; 2 3]
      B = [1 1; 1 1]
      C = A - B
      C
```

```
[89]: 2×2 Matrix{Int64}:
       3  8
       1  2
```

```
[90]: A = [4 9; 2 3]
      B = [1 1; 1 1]
      C = A * B
      C
```

```
[90]: 2×2 Matrix{Int64}:
       13  13
        5   5
```

Рис. 13: примеры операций над матрицами

```
[91]: A = [4 9; 2 3]
      D = 3
      C = A * D
      C
```

```
[91]: 2×2 Matrix{Int64}:
       12  27
        6   9
```

```
[92]: A
```

```
[92]: 2×2 Matrix{Int64}:
       4  9
       2  3
```

```
[93]: A'
```

```
[93]: 2×2 adjoint(::Matrix{Int64}) with eltype Int64:
       4  2
       9  3
```

```
[94]: using LinearAlgebra
      u = [3, 2, 1]
      v = [1, 4, 7]
      prod = dot(u,v)
```

```
[94]: 18
```

```
[95]: u = [3, 2, 1]
      v = [1, 4, 7]
      prod = cross(u,v)
```

```
[95]: 3-element Vector{Int64}:
        10
       -20
        10
```

Рис. 14: 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

1. JuliaLang [Электронный ресурс]. 2025 JuliaLang.org contributors. URL: https://julialang.org/ (дата обращения: 06.10.2025).
2. Julia 1.11 Documentation [Электронный ресурс]. 2025 JuliaLang.org contributors. URL: https://docs.julialang.org/en/v1/ (дата обращения: 06.10.2025).