

Компьютерный практикум по статистическому анализу данных

презентация к лабораторной работе №1

Julia. Установка и настройка. Основные принципы

Ким И. В. НФИбд-01-21

Российский университет дружбы народов, Москва, Россия

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 (разделы 1.3.1 и 1.3.2).
2. Используя Jupyter Lab, повторите примеры из раздела 1.3.3.
3. Выполните задания для самостоятельной работы (раздел 1.3.4).

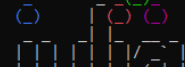
Задание для самостоятельной работы

Задание для самостоятельной работы

1. Изучите документацию по основным функциям Julia для чтения / записи / вывода информации на экран: `read()`, `readline()`, `readlines()`, `readdlm()`, `print()`, `println()`, `show()`, `write()`. Приведите свои примеры их использования, поясняя особенности их применения.
2. Изучите документацию по функции `parse()`. Приведите свои примеры её использования, поясняя особенности её применения.
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.
4. Приведите несколько своих примеров с пояснениями с операциями над матрицами и векторами: сложение, вычитание, скалярное произведение, транспонирование, умножение на скаляр.

Выполнение лабораторной работы

```
PS C:\WINDOWS\system32> julia
```



```
Documentation: https://docs.julialang.org
Type "?" for help, "]?" for Pkg help.

Version 1.10.5 (2024-08-27)
Official https://julialang.org/ release

julia>
```

Повторил примеры из раздела 1.3.3.

```
[2]: typeof(3), typeof(3.5), typeof(3/3.55), typeof(sqrt(3+4im)), typeof(pi)
[2]: (Int64, Float64, Float64, ComplexF64, Irrational{:π})

[3]: 1.0/0.0, 1.0/(-0.0), 0.0/0.0
[3]: (Inf, -Inf, NaN)

[4]: typeof(1.0/0.0), typeof(1.0/(-0.0)), typeof(0.0/0.0)
[4]: (Float64, Float64, Float64)

[5]: for T in [Int8, Int16, Int32, Int64, Int128, UInt8, UInt16, UInt32, UInt64, UInt128]
      println("$(lpad(T,7)): [$(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]
```

Повторил примеры из раздела 1.3.3.

```
[6]: Int64(2.0), Char(2), typeof(Char(2))
[6]: (2, '\x02', Char)
[7]: convert{Int64,2.0}, convert{Char,2}
[7]: (2, '\x02')
[8]: typeof(promote{Int8(1), Float16(4.5), Float32(4.1)})
[8]: Tuple{Float32, Float32, Float32}
[9]: function f(x)
      x^2
    end
[9]: f (generic function with 1 method)
10]: f(4)
10]: 16
11]: g(x)=x^2
11]: g (generic function with 1 method)
12]: g(8)
12]: 64
```

Повторил примеры из раздела 1.3.3.

```
[15]: a = [4 7 6]
      b = [1, 2, 3]
      a[2], b[2]
```

```
[15]: (7, 2)
```

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

```
[17]: 2x2 Matrix{Int64}:
      1  2
      3  4
```

```
[20]: Am[1,1], Am[1,2], Am[2,1], Am[2,2]
```

```
[20]: (1, 2, 3, 4)
```

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

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

```
[28]: aa, AA, aa'
```

```
[28]: ([1 2], [1 2; 3 4], [1; 2;:])
```

Задания для самостоятельной работы

Задание №1

Прикрепил к проекту txt файл “try.txt” и прочитал его с помощью функции read()

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

```
"Hello World!"
```

Функция `readline()` читает первую строку

```
readline("try.txt")
```

```
"Hello World!"
```


Функция `readlines()` считывает все строки из текстового файла

```
readlines("try.txt")
```

```
3-element Vector{String}:  
 "Hello World!Hello World!Hello World!"  
 "awdawdawdwHello World!"  
 "Hello World!"
```

Функция `print()` выводит текст без перехода на новую строку

```
print("Hello")
```

```
print("Hello")
```

```
HelloHello
```

Функция `println()` выводит текст с переходом на новую строку

```
println("Hello")  
println("Hello")
```

Hello

Hello

Функция `show()` выводит все, что находится в скобках как оно есть

```
show("Hello World")
```

```
show([123])
```

```
"Hello World"[123]
```

С помощью функции `write()` можно записать текст в открытый файл(нужно указать режим открытия)

```
: io = open("try.txt","a")
  write(io, "Hello World!\n")
  close(io)

: readline("try.txt")

: "Hello World!Hello World!Hello World!"
```

Задание №2

Функция `parse()` позволяет преобразовать(распарсить) строку в числа или выражение

```
x=readline()
y=readline()
z=x*y
```

```
stdin> 1
stdin> 2
"12"
```

```
x=Meta.parse(readline())
y=Meta.parse(readline())
z=x*y
```

```
stdin> 1
stdin> 2
2
```

```
x = "6*7"
eval(Meta.parse(x))
```

Задание №3

Базовые математические операции

```
[ ]: sum = 5+3
```

```
[ ]: 8
```

```
[ ]: difference = 5-3
```

```
[ ]: 2
```

```
[ ]: product = 5*3
```

```
[ ]: 15
```

```
[ ]: quotient = 5/3
```

```
[ ]: 1.6666666666666667
```

```
[ ]: power = 5 ^ 3
```

```
[ ]: 125
```

```
[ ]: modulus = 5 % 3
```

```
[ ]: 2
```

Базовые математические операции

```
√5
```

```
2.23606797749979
```

```
5÷3
```

```
1
```

```
5==3
```

```
false
```

```
5!=3
```

```
true
```

```
5==3 && 5!=3
```

```
false
```

```
5==3 || 5!=3
```

```
true
```

```
5<3
```

```
false
```

```
5>=3
```

```
true
```

Задание №4

Операции над матрицами

```
a=[5 3 2]  
b=[1 2 3]  
c= [5 5 ; 3 2; 3 4 ]
```

```
3x2 Matrix{Int64}:  
 5  5  
 3  2  
 3  4
```

```
a+b
```

```
1x3 Matrix{Int64}:  
 6  5  5
```

```
a-b
```

```
1x3 Matrix{Int64}:  
 4  1 -1
```

```
a*c
```

```
1x2 Matrix{Int64}:  
40 39
```

Операции над матрицами

```
d=[1 1]
k=[3 3; 4 4]
```

```
2x2 Matrix{Int64}:
 3  3
 4  4
```

```
d*k*d'
```

```
1x1 Matrix{Int64}:
 14
```

```
d,k,d'
```

```
([1 1], [3 3; 4 4], [1; 1;:])
```

```
d=d*4
```

```
1x2 Matrix{Int64}:
 4  4
```

```
d
```

```
1x2 Matrix{Int64}:
 4  4
```

Выводы

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