

Защита лабораторной работы № 1

Julia. Установка и настройка. Основные принципы.

Компьютерный практикум по статистическому анализу данных

Работу Выполнил:
Саинт-Амур Измаэль
Группа: НПИбд-01-20

Цель работы

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

Установка под операционную систему Julia, Jupyter

я установил Шоколадный
(<https://chocolatey.org/>) устанавливается
через административную оболочку. Затем
установить Far Manager, Notepad++, Julia
через этот менеджер, дистрибутив Anaconda
(Python 3.x).

```
PS C:\Users\Scorpion 1.0> julia
```



```
| Documentation: https://docs.julialang.org  
|  
| Type "?" for help, "]?" for Pkg help.  
|  
| Version 1.9.3 (2023-08-24)  
| Official https://julialang.org/ release  
  
julia>
```

Изучаем 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

read(): Эта функция позволяет считывать данные из файла или потока. Вы можете считать данные в различных форматах, таких как текстовые данные, числа и т. д

```
In [20]: file = open("C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статистическому анализу данных\\Lab1\\Lab1.txt", "r") #  
data = read(file, String) # Считываем строку из файла  
close(file) # Закрываем файл  
println(data)
```

Family: The Heartbeat of Life

Family is the cornerstone of our existence, the bedrock upon which our journey through life is built.
It is a tapestry woven with the threads of love, trust, and shared experiences.
In the embrace of family, we find comfort, support, and an unwavering sense of belonging.

Each member is a unique piece of the puzzle, contributing to the beautiful mosaic that is our family.
From the laughter that echoes through the halls to the quiet moments of understanding, familial bonds are the ties that bind us together.
It is a haven where we celebrate successes, weather storms, and grow together.

In the tapestry of family, there are moments of joy, challenges that test our strength, and the constant rhythm of change.
Yet, through it all, the love we share remains a constant, an anchor that grounds us in the turbulent sea of life.

Our family is a source of inspiration, a refuge in times of need, and a source of enduring love.
It is in the warm embrace of family that we discover the true meaning of home, where acceptance and understanding form the foundation of our collective journey.

Изучаем документацию по основным функциям Julia

• *readline()*: Эта функция считывает одну строку из файла или потока.

• *print()* и *println()*: Эти функции используются для вывода текста в стандартный вывод. *print()* выводит текст без перевода строки, в то время как *println()* добавляет перевод строки в конце.

```
In [21]: file = open("C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статистике\\data\\family.txt")
line = readline(file) # Считываем первую строку из файла
close(file) # Закрываем файл
println(line)
```

Family: The Heartbeat of Life

```
In [26]: x = 42
y = "Hello, World!"

print("Значение x: ", x, ", Значение y: ", y)
println("Это новая строка.")
```

Значение x: 42, Значение y: Hello, World!Это новая строка.

Функция parse()

• Функция *parse()* в языке программирования Julia используется для преобразования строковых представлений данных в их соответствующие типы данных. Это может быть полезно, например, при считывании данных из файлов или пользовательского ввода, где данные сначала считываются как строки, а затем преобразуются в нужные типы данных. Вот примеры использования функции *parse()*:

Преобразование строки в целое число (Int):

```
In [35]: str = "54"
          x = parse{Int, str}
          println(typeof(x))
          println(x)

          Int64
          54
```

- Преобразование строки в число с плавающей запятой (Float64):

```
str = "54.0"
y = parse{Float64, str}
println(typeof(y))
println(y)

Float64
54.0
```

- Преобразование строки в булево значение (Bool):

```
str = "true"
z = parse{Bool, str}
println(typeof(z))
println(z)

Bool
true
```


синтаксис Julia для базовых математических операций

Сложение, вычитание, умножение и деление

Возведение в степень и извлечение корня:

Сравнение: Сравнение операндов возвращает логические значения true (истина) или false (ложь).

Логические операции: Логические операции выполняются над булевыми значениями true и false.

```
: p = true  
  q = false  
  # Логическое И (AND)  
  result = p && q
```

```
: false
```

```
In [48]: a = 6  
        b = 9  
        # Сложение  
        c = a + b
```

```
Out[48]: 15
```

```
In [49]: a = 6  
        b = 9  
        # вычитание  
        c = a - b
```

```
Out[49]: -3
```

```
In [51]: a = 6  
        b = 9  
        # умножение  
        c = a * b
```

```
Out[51]: 54
```

```
In [52]: a = 6  
        b = 9  
        # деление  
        c = a / b
```

```
Out[52]: 0.6666666666666666
```

```
In [53]: x = 4  
        square = x^2
```

```
Out[53]: 16
```

```
In [54]: x = 4  
        square = sqrt(x)
```

```
Out[54]: 2.0
```

```
In [55]: a = 9  
        b = 8  
        result = a > b
```

```
Out[55]: true
```

```
In [56]: a = 9  
        b = 8  
        result = a < b
```

```
Out[56]: false
```

```
In [57]: a = 9  
        b = 8  
        result = a == b
```

```
Out[57]: false
```

Приведем несколько своих примеров с операциями над матрицами и векторами

Сложение и вычитание векторов: Векторы можно складывать и вычитать покомпонентно, предполагая, что векторы имеют одинаковую длину.

```
In [75]: using LinearAlgebra

a = [1, 2, 3]
b = [4, 5, 6]
# Сложение векторов
result = a + b
```

```
Out[75]: 3-element Vector{Int64}:
 5
 7
 9
```

соответствующих элементов двух век

Скалярное произведение (скалярное умножение) векторов: Скалярное произведение векторов вычисляется как сумма произведений соответствующих элементов двух векторов.

```
In [72]: using LinearAlgebra

a = [1, 2, 3]
b = [4, 5, 6]
result = dot(a, b)
```

```
Out[72]: 32
```

транспонирование матрицы:

Транспонирование меняет строки и столбцы матрицы местами.

```
78]: A = [1 2 3; 4 5 6; 7 8 9]
# Транспонируем матрицу A
M_transpose = transpose(A)
```

```
78]: 3x3 transpose(::Matrix{Int64}) with eltype Int64:
 1  4  7
 2  5  8
 3  6  9
```


- Успешная Установка и Подготовка Рабочего Пространства:
успешно установили Шоколадный, Julia, Jupyter, Far Manager, Notepad++ и подготовили свое рабочее пространство для работы.
- Знакомство с Основами Синтаксиса Julia:
я ознакомился с основами синтаксиса Julia, такими как переменные, операции, функции, массивы и работа с файлами.
- Использование Основных Функций:
я использовал базовые функции Julia для чтения и записи данных, вывода на экран, выполнения математических операций и т.д. Примерами могут быть `read()`, `write()`, `println()`, `readdlm()` и другие.