РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

ОТЧЕТ

по лабораторной работе № 1

**Julia. Установка и настройка. Основные**

**принципы.**

*дисциплина: Компьютерный практикум по статистическому анализу данных*

Студент: Ким Реачна

Группа: НПИбд-01-20

Студенческий билет: 1032205204

**МОСКВА**

2022 г.

# Постановка задачи

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

# Выполнение работы

Выполнила установку Far Manager, Notepad++, Julia и Anaconda Distribution через менеджер пакетов Chocolatey (Рис. 1). А также ознакомилась с пунктом «Основы работы в блокноте Jupyter». Попробовала выполнить простейшие примеры и задействовала предложенные горячие клавиши для более комфортной работы со средой (Рис. 2).

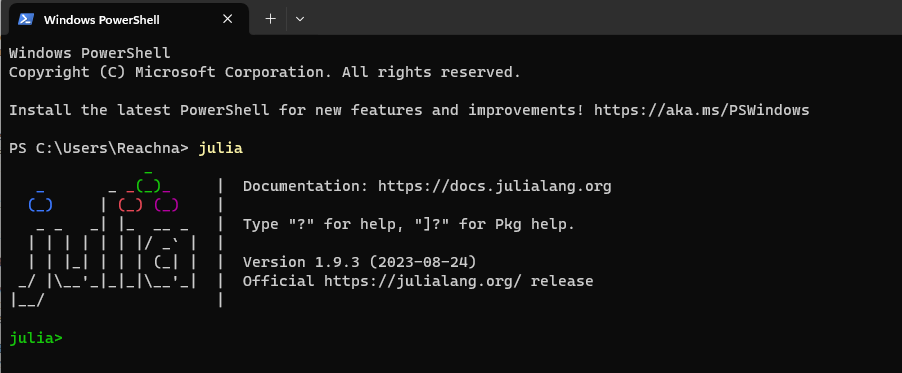


Рис. 1 Установка пакетов для работы с Jupyter



Рис. 2 Простейшие операции на языке Julia в Jupyter Lab

Чтобы получить информацию о незнакомой функции в Julia, поставьте вопросительный знак в ячейке перед названием этой функции (рис. 2): ?println()

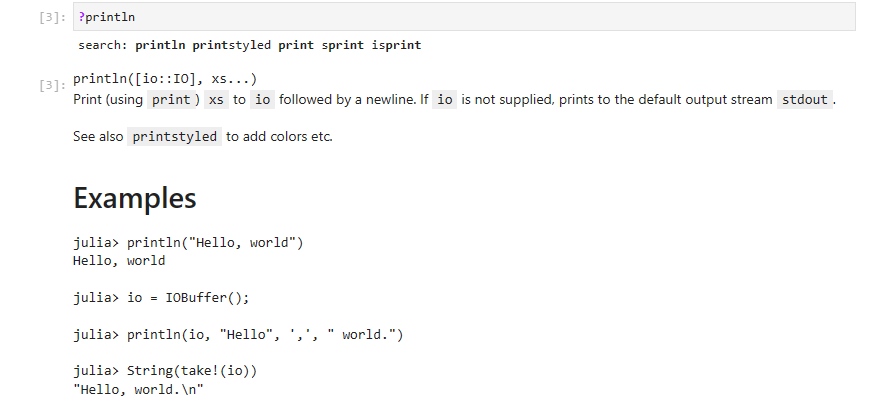


Рис. 3 Пример получения информации по функции println на языке Julia

**Основы синтаксиса Julia на примерах**

* Определение типа числовой величины (Рис. 4):

typeof(Number)- здесь Number — конкретное число.

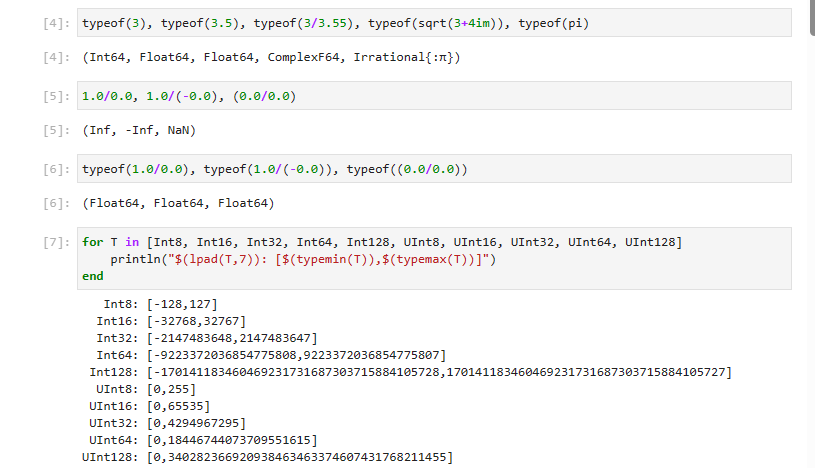


Рис. 4 Примеры определения типа числовых величин

* Разные способы для преобразования типов данных на языке Julia (Рис. 5)

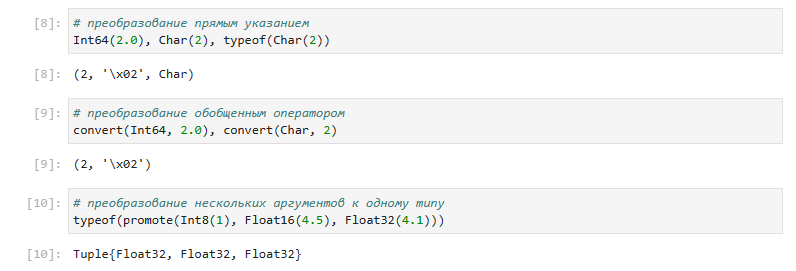


Рис. 5 Примеры приведения аргументов к одному типу

* Базовый синтаксис определения функции (Рис. 6):

Function <Имя>(<СписокПараметров>)

<действия>

End



Рис. 6 Примеры определения функций

* Работы с массивами

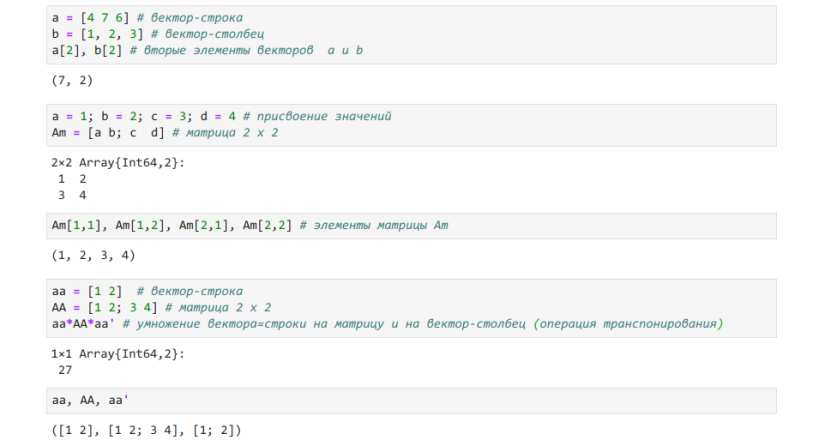


Рис. 7 Примеры работы с массивами

**Задания 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.

Для каждой функции в первой ячейке я вывожу информацию о ней (с помощью вопросительного знака) (Рис.8).



Рис. 8 Получения информации по функции

* **read():** данная функция считывает значение типа T из io, в общепринятом двоичном представлении.

Пример: Выведем строку “Hello, world!”

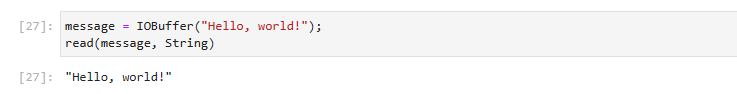


Рис. 9 Пример использования read()

* **readline()**: считывание **одной строки** текста из данного потока ввода-вывода или файла. Строки на входе в файле заканчиваются буквой "\n " или "\r\n " или концом входного потока. Если keep имеет значение false (по умолчанию), эти конечные символы новой строки удаляются из строки до ее возврата. Когда keep имеет значение true, они возвращаются как часть строки.



Рис. 10 myfile.txt

Как мы видим в начале я открыла текстовый файл на запись и поместила в него строку “Hello world and Julia!” и в конце указала ‘\n’. Функция write выводит нам на количество байтов, записанных в поток.. Далее с помощью функции readline() я попробовала 2 способа вывода записи, без параметра keep и с ним. В итоге я получила следующее:

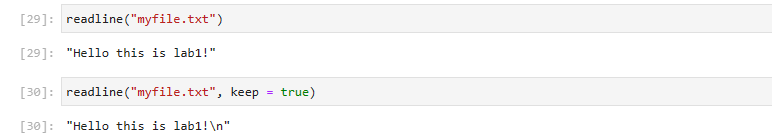


Рис. 11 Пример использования readline()

* **readlines()**: Принципиальное отличие от предыдущей функции – считывание **всех строк** из файла или потока ввода-вывода. В данной функции также присутствует параметр keep и выполняет те же самые операции. Сохранение строк из файла происходит как сохранение вектора строк. Создала и записала какой-то текст в файл my\_file.txt (Рис. 12).

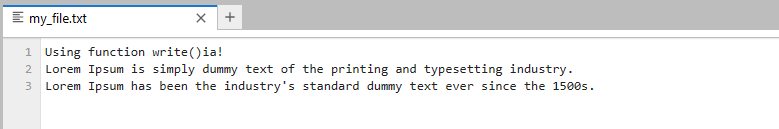


Рис. 12 фалй my\_file.txt

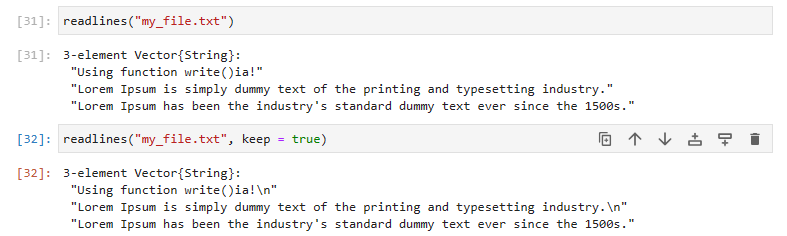


Рис. 13 Пример использования readlines()

* **readdlm()**: Функция readdlm(). Для того, чтобы воспользоваться данной функцией воспользовалась утилитой DelimitedFiles для чтения и записи файлов с разделителями. Предполагается, что столбцы разделены одним или несколькими пробелами. В качестве разделителя конца строки принимается \ n.

Также при вызове функции указывается тип данных в который преобразуется передаваемая строка. Ниже я создала файл с числами разделенными точкой с запятой, и далее я считала и преобразовала к Int32 и Float32.



Рис. 14 Фалй delimited\_file.txt

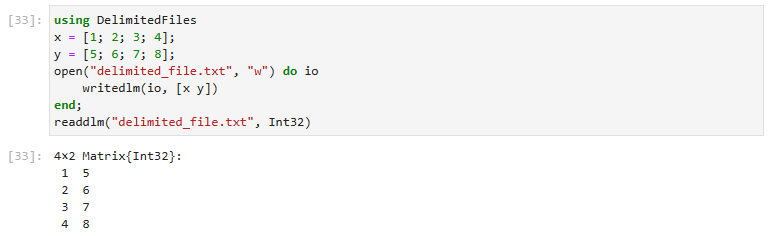


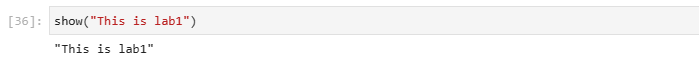
Рис. 15 Пример использованияя readdlm()

* **print()** и **println()**: Запись происходит в io (или в стандартный выходной поток stdout, если io не задан) общепринятое (недекорированное) текстовое представление. Представление, используемое print, включает минимальное форматирование и пытается избежать специфичных для Julia деталей. Функция println(). Эквивалентен функции print() только после вывода переходит на следующую строку



Рис. 16 Пример использования println()

* **show()**: записывает текстовое представление значения x в выходной поток ввода-вывода. Новые типы T должны перегружать функцию show (io:: IO, x:: T). Представление, используемое show, как правило, включает в себя специфичное для Julia форматирование и информацию о типе.

****

* **write():** записывает общепринятое двоичное представление значения в данный поток ввода-вывода или файл. Возвращает количество байтов, записанных в поток.



Рис. 17 Пример использования write()



Рис. 18 Результат отображается в файле myfile.txt

1. Изучите документацию по функции parse(). Приведите свои примеры её использования, поясняя особенности её примененя.

Парсит строку в число. Для целочисленных типов можно указать base (по умолчанию 10). Для типов с плавающей запятой строка анализируется как десятичное число с плавающей запятой. Комплексные числа разбираются из десятичных строк формы «R ± Iim» как Комплекс (R, I) запрошенного типа; «i» или «j» также могут использоваться вместо «im», также разрешены «R» или «Iim». Если строка не содержит допустимого числа, возникает ошибка.

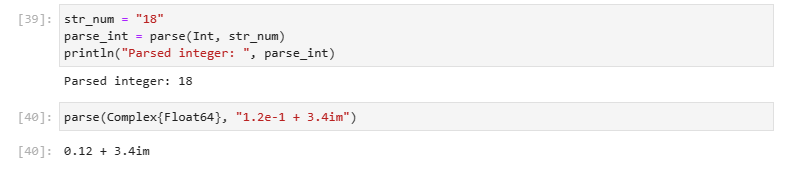


Рис. 19 Пример использования parse()

1. 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.

Объявите 2 переменные a = 3 и b = 4, затем вычислите арифметические операторы (+, -, \*, /, ^):

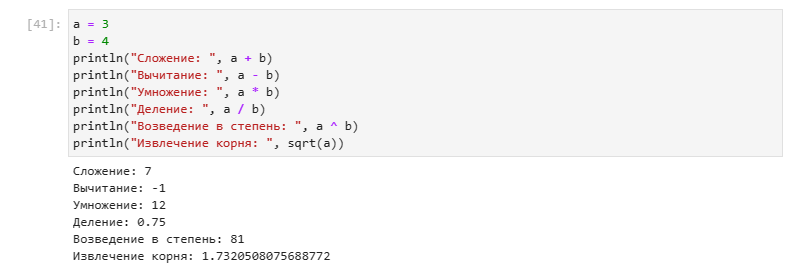


Рис. 20 Примеры арифметических операций с данными

Объявите 2 переменные x = 5 и = 6, затем выполните сравнение:

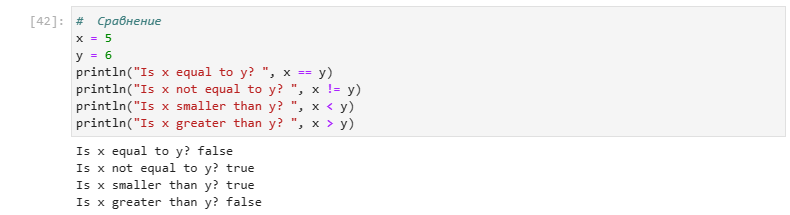


Рис. 21 Пример сравнение

Логические операции:

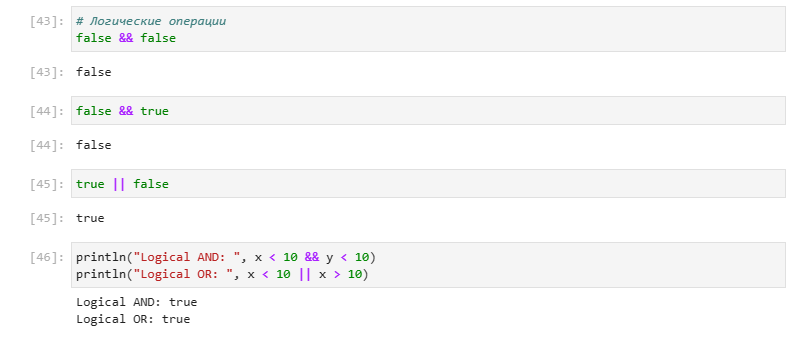


Рис. 22 Пример Логические операции

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

Создала 2 матрицы одинакового размера 3x3 для удобства.

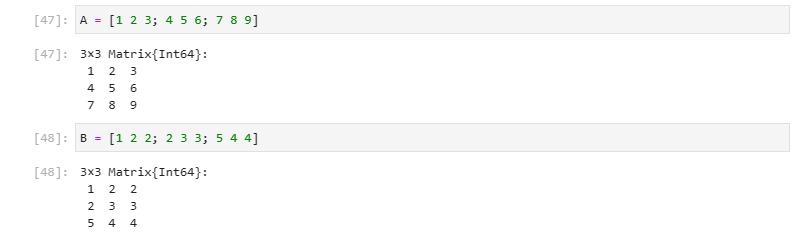


Рис. 23 Создайте 2 матрицы 3 x 3

Далее идут привычные нам операции – сложение (+), вычитание (-), скалярное произведение, и транспонирование (‘).



Рис. 24 Основные операции с матрицами и векторами

**Вывод:**

Подготовила рабочее пространство и инструментарии для работы с языком программирования Julia, на простейших примерах познакомилась с основами синтаксиса Julia.