**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

# ОТЧЕТ
# ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

*дисциплина: Компьютерный практикум*

*по статистическому анализ данных*

Студент: Доре Стевенсон Эдгар

Группа: НКН-бд-01-19

**МОСКВА**

2022 г.

## Постановка задачи

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

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).

## Выполнение работы

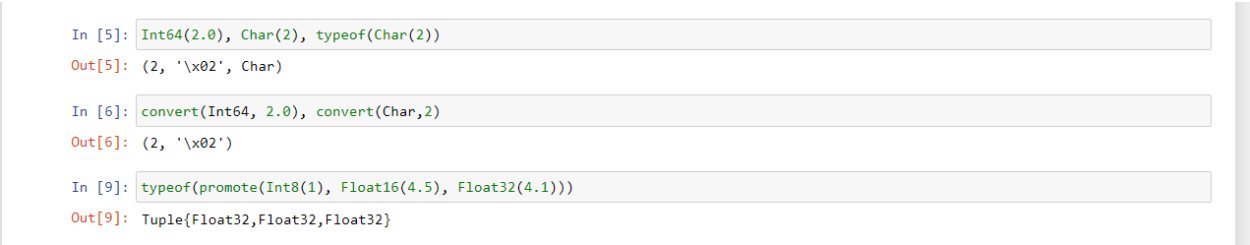
1. Установите под свою операционную систему Julia, Jupyter (разделы 1.3.1 и 1.3.2).

Посмотрели ознакомительное видео и выполнили установку требуемых приложений

2. Используя Jupyter Lab, повторите примеры из раздела 1.3.3.

Получим минимальные и максимальные значения целочисленных типов


Примеры приведения типов




Примеры работы с функциями

```
Out[10]: f (generic function with 1 method)

In [11]: f(4)
Out[11]: 16

In [12]: g(x)=x^2
Out[12]: g (generic function with 1 method)

In [13]: g(8)
Out[13]: 64
```

Примеры работы с матрицами

```
In [14]: a = [4 7 6]
         b = [1, 2, 3]
         a[2], b[2]
Out[14]: (7, 2)

In [15]: a=1; b=2; c=3; d=4
         Am = [a b; c d]
Out[15]: 2×2 Array{Int64,2}:
          1  2
          3  4

In [16]: Am[1,1], Am[1,2], Am[2,1], Am[2,2]
Out[16]: (1, 2, 3, 4)

In [17]: aa = [1 2]
         AA = [1 2; 3 4]
         aa*AA*aa'
Out[17]: 1×1 Array{Int64,2}:
          27

In [18]: aa, AA, aa'
Out[18]: ([1 2], [1 2; 3 4], [1; 2])
```

3. Выполните задания для самостоятельной работы (раздел 1.3.4).

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

- read() – считывает значение указанного типа в бинарном представлении read(stream, type)

```
In [4]: io=IOBuffer("123")
        c=read(io)
        c
Out[4]: 3-element Array{UInt8,1}:
         0x31
         0x32
         0x33

In [6]: io=IOBuffer("123")
        c=read(io, String)
        c
Out[6]: "123"
```

- readline() – считывает строку до символа завершения строки, включая

этот символ

```
In [49]: readline("file.txt", keep=true)
Out[49]: "1 234\n"
```

```
In [46]: str=readline()
         str

         stdin> 123
Out[46]: "123"
```

- readlines() – считывает все строки, поданные на вход в виде массива

```
In [52]: c1=readlines("file.txt", keep=true)
Out[52]: 2-element Array{String,1}:
          "1 234\n"
          "5 6 7\n"
```

- print() – распечатывает на экран содержимое переменной

```
In [55]: print(c1)
         print(c1)
         ["1 234\n", "5 6 7\n"]["1 234\n", "5 6 7\n"]
```

- println() – распечатывает на экран с новой строки содержимое переменной

```
In [56]: println(c1)
         println(c1)
         ["1 234\n", "5 6 7\n"]
         ["1 234\n", "5 6 7\n"]
```

- show() – распечатывает на экран содержимое переменной, для сложных выражений выполняет форматирование, можно указать поток вывода

```
In [59]: show("c1")
         "c1"
```

- write() – выполняет запись содержимого переменной в указанный поток, возвращает число записанных байт

```
In [64]: io=open("wfile.txt", "w")
         write(io, c)
Out[64]: 11
```

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

Функция parse() преобразовывает строку в численный тип, можно указать основание системы счисления и поток ввода.

```
In [68]: a=parse(Int64,c)
Out[68]: 123
```

```
In [13]: a=parse(Int, "1234")
Out[13]: 1234
```

```
In [16]: a=parse(Int, "123", base=4)
Out[16]: 27
```

```
In [18]: a=parse(Float32, "12.3")
Out[18]: 12.3f0
```

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.

Синтаксис предполагает использование стандартный символов в качестве обозначения операций

| | |
|---|---|
| + | Сложение |
| - | Вычитание, унарный минус |
| * | умножение |
| / | правостороннее деление |
| \ | левостороннее деление |
| ^ | степень, работает также для матриц |
| fma() | вычисление без округления |
| inv() | обратное число |
| div() | целочисленное деление |
| mod() | остаток от деления |
| == | сравнение на равенство |
| != | сравнение на неравество |
| < | сравнение на меньшее |
| > | сравнение на большее |
| <= | меньшее либо равное |
| >= | большее либо равное |
| ~ | побитовое НЕ |
| & | побитовое И |
| \| | побитовое ИЛИ |
| ! | логическое НЕ |
| xor() | побитовое ИСКЛ ИЛИ |
| && | логическое И |
| \|\| | логическое ИЛИ |
| sin(), cos(), tan() | тригонометрические в радианах |
| sind(), cosd(), tand() | тригонометрические в градусах |
| sinh(), cosh(), tanh() | гиперболические тригонометрические функции |

Для всех тригонометрических функций реализованы обратные функции

| | |
|---|---|
| log(x) | натуральный логарифм |
| log(b, x) | логарифм по основанию b |
| exp() | экспонента e^x |

| | |
|---|---|
| exp2() | 2^x |
| exp10() | 10^x |
| round() | математическое округление |
| abs() | модуль |
| abs2() | квадрат модуля |
| sqrt() | квадратный корень |
| cbrt() | кубический корень |
| factorial() | факториал |

```
In [21]: 1.2+3
Out[21]: 4.2

In [22]: +(3,2)
Out[22]: 5

In [23]: 4/2
Out[23]: 2.0

In [24]: 2/4
Out[24]: 0.5

In [25]: 4\2
Out[25]: 0.5

In [26]: 2\4
Out[26]: 2.0

In [27]: div(5,2)
Out[27]: 2

In [28]: mod(5,2)
Out[28]: 1

In [29]: 3.0==3
Out[29]: true

In [30]: 3.0<3
Out[30]: false
```

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

```
In [38]: M=[1 2 5; 3 4 7; 4 6 9]
         N=[3 2 4; 6 8 9; 4 3 2]
         println(M+N)
         println(M-N)
         println(M*N)
         println(M')

         [4 4 9; 9 12 16; 8 9 11]
         [-2 0 1; -3 -4 -2; 0 3 7]
         [35 33 32; 61 59 62; 84 83 88]
         [1 3 4; 2 4 6; 5 7 9]

In [39]: print(4*N)

         [12 8 16; 24 32 36; 16 12 8]
```