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 математическому моделированию

Студент: Саинт-Амур Измаэль

Группа: НПИбд-01-20

МОСКВА

2023 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.

Выполнение работы

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

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

Рис. 1.1. Установка пакетов для работы с Jupyte

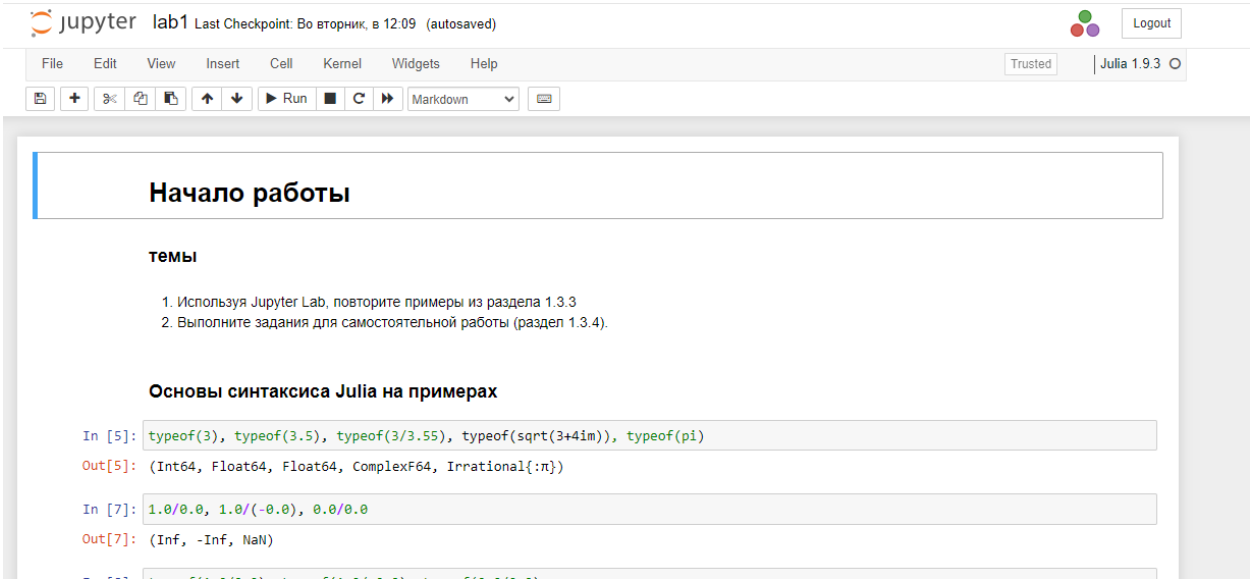

Рис. 1.2. Простейшие операции на языке Julia в Jupyter Lab

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

1.3.3. Основы синтаксиса Julia на примерах

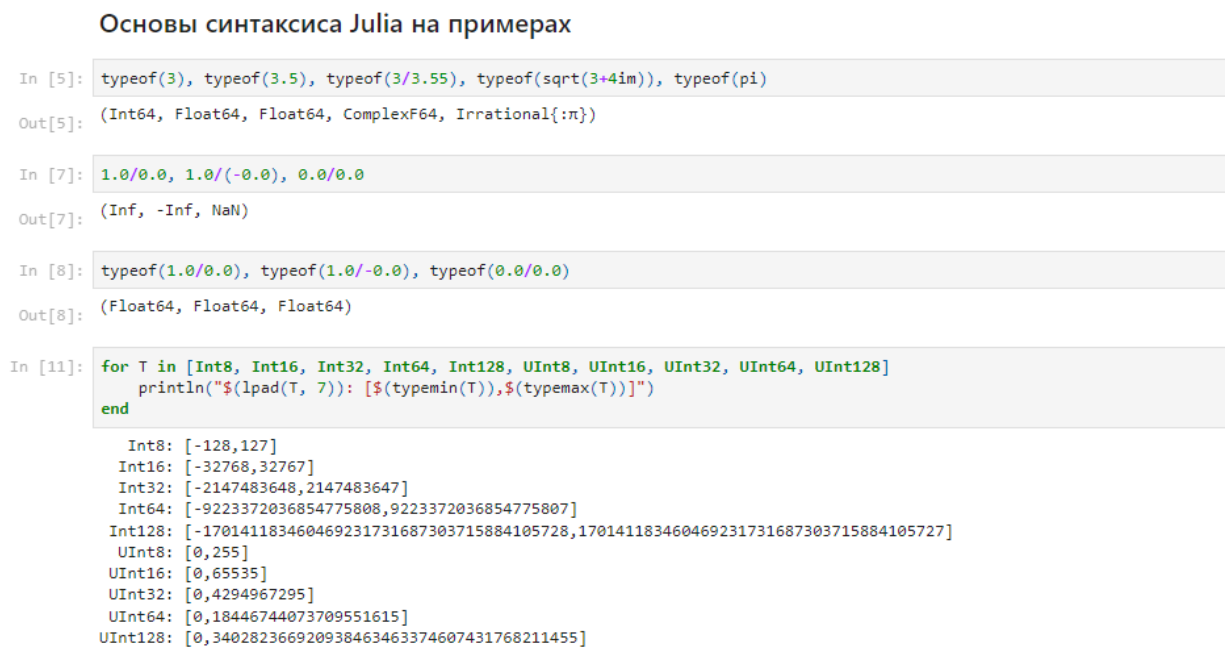

Рис. 1.3. Примеры определения типа числовых величи

Рис. 1.4. Примеры приведения аргументов к одному типу

```
In [18]: function f(x)
             x^2
         end

Out[18]: f (generic function with 1 method)

In [19]: f(4)

Out[19]: 16

In [20]: g(x)=x^2

Out[20]: g (generic function with 1 method)

In [21]: g(8)

Out[21]: 64
```

Рис. 1.5. Примеры определения функций

```
In [22]: a = [4 7 6] # вектор-строка
         b= [1, 2, 3] #вектор-столбец
         a[2], b[2] #вторые элементы векторов a и b

Out[22]: (7, 2)

In [24]: a = 1; b = 2; c = 3; d = 4 #присвоение значений
         Am = [a b; c d] # матрица 2 x 2

Out[24]: 2×2 Matrix{Int64}:
          1  2
          3  4

In [25]: Am[1,1], Am[1,2], Am[2,2] # элементы матрицы Am

Out[25]: (1, 2, 4)

In [26]: aa = [1 2] # вектор-строка
         AA = [1 2; 3 4] # матрица 2 x 2
         aa*AA*aa' # умножение вектора=строки на матрицу и вектор-столбец  (операция транспонирования)

Out[26]: 1×1 Matrix{Int64}:
          27

In [27]: aa, AA, aa'

Out[27]: ([1 2], [1 2; 3 4], [1; 2;;])
```

Рис. 1.6. Примеры работы с массивами

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

1.3.4. 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

```
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 foun
dation of our collective journey.
```

```
In [21]: file = open("C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статисти
         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.
```

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

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 булевое значение (Bool):

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

```
Bool
true
```

4. 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.

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 [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

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

Out[53]: 16

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

Out[54]: 2.0
```

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

```
n [75]: using LinearAlgebra

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

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

соответствующих элементов двух век

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

t[72]: 32
```

Выводы

- 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 ознакомились 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.

Начало работы

темы

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).

1.3.3. Основы синтаксиса Julia на примерах

```
In [1]: typeof(3), typeof(3.5), typeof(3/3.55), typeof(sqrt(3+4im)), typeof(pi)
Out[1]: (Int64, Float64, Float64, ComplexF64, Irrational{:π})

In [2]: 1.0/0.0, 1.0/(-0.0), 0.0/0.0
Out[2]: (Inf, -Inf, NaN)

In [3]: typeof(1.0/0.0), typeof(1.0/-0.0), typeof(0.0/0.0)
Out[3]: (Float64, Float64, Float64)

In [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]
```

Рис. 1.5. Примеры определения типа числовых величин

```
In [6]: Int64(2.0), Char(2), typeof(Char(2))
Out[6]: (2, '\x02', Char)

In [7]: convert(Int64, 2.0), convert(Char,2)
Out[7]: (2, '\x02')

In [8]: typeof(promote(Int8(1), Float16(4.5), Float32(4.1)))
Out[8]: Tuple{Float32, Float32, Float32}
```

Рис. 1.6. Примеры приведения аргументов к одному типу

```
In [9]: function f(x)
            x^2
        end

Out[9]: f (generic function with 1 method)

In [10]: f(4)

Out[10]: 16

In [11]: g(x)=x^2

Out[11]: g (generic function with 1 method)

In [12]: g(8)

Out[12]: 64
```

Рис. 1.7. Примеры определения функций

```
In [13]: a = [4 7 6] # вектор-строка
         b = [1, 2, 3] # вектор-столбец
         a[2], b[3] # вторые элементы векторов a и b

Out[13]: (7, 3)

In [14]: a = 1; b = 2; c = 3; d = 4 # присвоение значений
         Am = [a b; c d] # матрица 2 x 2

Out[14]: 2×2 Matrix{Int64}:
          1  2
          3  4

In [15]: Am[1,1], Am[1,2], Am[2,1], Am[2,2] #  элементы матрицы Am

Out[15]: (1, 2, 3, 4)

In [16]: aa = [1 2] # вектор-строка
         AA = [1 2; 3 4] # матрица 2 x 2
         aa*AA*aa' #  умножение вектора=строки на матрицу и вектор-столбец  (операция транспонирования

Out[16]: 1×1 Matrix{Int64}:
          27

In [17]: aa, AA, aa'

Out[17]: ([1 2], [1 2; 3 4], [1; 2;;])
```

Рис. 1.8. Примеры работы с массивами

1.3.4. 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

- *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.

Пример использования read():

In [20]:
```
file = open("C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статистическому анал
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 ou
r family.
From the laughter that echoes through the halls to the quiet moments of understanding, famil
ial 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 l
ove.
It is in the warm embrace of family that we discover the true meaning of home, where accepta
nce and understanding form the foundation of our collective journey.


In the grand symphony of life, family is the melody that resonates in our hearts,
a melody that continues to play, connecting generations and weaving the story of who we are.
```

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

Пример использования readline():

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

```
Family: The Heartbeat of Life
```

- *readlines()*: Эта функция считывает все строки из файла и возвращает их в виде массива строк.

Пример использования readlines():

In [23]:
```
file = open("C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статистическому анал
lines = readlines(file) # Считываем все строки из файла
close(file) # Закрываем файл
for line in lines
    println(line)
end
```

```
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 ou
r family.
From the laughter that echoes through the halls to the quiet moments of understanding, famil
ial 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 l
ove.
It is in the warm embrace of family that we discover the true meaning of home, where accepta
nce and understanding form the foundation of our collective journey.


In the grand symphony of life, family is the melody that resonates in our hearts,
a melody that continues to play, connecting generations and weaving the story of who we are.
```

- *readdlm()*: Эта функция считывает данные из файла, используя разделитель (по умолчанию это запятая) и возвращает их в виде массива.

Пример использования readdlm()

In [25]:
```
using DelimitedFiles

file_path = "C:\\Users\\Scorpion 1.0\\Desktop\\Компьютерный практикум по статистическому анал

# Read data from the CSV file
data = readdlm(file_path, ',')

# Display the data
println(data)
```

```
Any["Nom" "Prénom" "Age"; "Doe" "John" 30; "Smith" "Jane" 25; "Brown" "Michael" 35]
```

- *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.

Пример использования print() и println():

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.
```

- *show()*: Эта функция используется для вывода информации о объекте, как он был бы отображен в интерактивной оболочке Julia. Это полезно для отладки и исследования данных.

Пример использования show():

In [27]:
```
x = [3*4, 9.0+4, 9-8]
show(x)
```

```
[12.0, 13.0, 1.0]
```

- *write()*: Эта функция используется для записи данных в файл или поток.

Пример использования write():

In [34]:
```
# Text data to be written to the file
text_data = "Hello, I'am Ismael From Haiti."

# Open the file for writing
file = open(file_path, "w")

# Write the text data to the file
write(file, text_data)
```

Out[34]:
```
30
```

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

- *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):

In [36]:
```
str = "54.0"
y = parse(Float64, str)
println(typeof(y))
    println(y)
```

```
Float64
54.0
```

- Преобразование строки в булевое значение (Bool):

In [37]:
```
str = "true"
z = parse(Bool, str)
println(typeof(z))
    println(z)
```

```
Bool
true
```

- Преобразование строки в дату (используя библиотеку Dates):

In [46]:
```
using Dates
date_str = "1998-08-31"
date = parse(Date, date_str)
println(typeof(date))
println(date)
```

```
Date
1998-08-31
```

- Обработка исключений при неправильном формате строки:

In [47]:
```
str = "не целое число"
try
    x = parse(Int, str)
    println(x)
catch e
    println("Ошибка: ", e)
end
```

```
Ошибка: ArgumentError("invalid base 10 digit 'н' in \"не целое число\"")
```

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.

- Сложение, вычитание, умножение и деление

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
```

- Возведение в степень и извлечение корня:

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

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

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

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

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

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
```

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

In [58]:
```
p = true
q = false
# Логическое И (AND)
result = p && q
```

Out[58]:
```
false
```

In [59]:
```
p = true
q = false
# Логическое ИЛИ (OR)
result = p || q
```

Out[59]:
```
true
```

In [63]:
```
p = true
q = false
# Логическое НЕ (NOT)
result = ! p
```

Out[63]:
```
false
```

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

- 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
```

In [73]:
```
using LinearAlgebra
a = [1, 2, 3]
b = [4, 5, 6]
# Вычитание векторов
result = a - b
```

Out[73]:
```
3-element Vector{Int64}:
 -3
 -3
 -3
```

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.

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

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

- Умножение матрицы на скаляр: Умножение матрицы на скаляр выполняется покомпонентно, где каждый элемент матрицы умножается на скаляр.

In [80]:
```
A = [1 2; 3 4]
scalar = 2
# Умножение матрицы A на скаляр
result = A * scalar
```

Out[80]:
```
2×2 Matrix{Int64}:
 2  4
 6  8
```

In [ ]: