Отчёт по лабораторной работе №2: Шифры перестановки

Дисциплина: Математические основы защиты информации и информационной безопасности

Савченко Елизавета Николаевна

Содержание

[1 Общая информация о задании лабораторной работы 3](#_Toc209895012)

[1.1 Цель работы 3](#_Toc209895013)

[1.2 Задание [1] 3](#_Toc209895014)

[2 Теоретическое введение [2] 3](#_Toc209895015)

[2.1 Шифры и симметричные шифры 3](#_Toc209895016)

[3 Выполнение лабораторной работы [1] 3](#_Toc209895017)

[3.1 Шифр 1 3](#_Toc209895018)

[3.2 Шифрование с помощью решёток 10](#_Toc209895019)

[3.3 Таблицы Виженера 12](#_Toc209895020)

[4 Выводы 19](#_Toc209895021)

[Список литературы 19](#_Toc209895022)

# 1 Общая информация о задании лабораторной работы

## 1.1 Цель работы

Ознакомиться с классическими примерами шифров перестановки.

## 1.2 Задание [1]

1. Реализовать шифры из задания.

# 2 Теоретическое введение [2]

## 2.1 Шифры и симметричные шифры

Первоначальное сообщение от одного пользователя к другому названо исходным текстом; сообщение, передаваемое через канал, названо зашифрованным текстом. Чтобы создать зашифрованный текст из исходного текста, отправитель использует алгоритм шифрования и совместный ключ засекречивания. Для того чтобы создать обычный текст из зашифрованного текста, получатель использует алгоритм дешифрования и тот же секретный ключ. Мы будем называть совместное действие алгоритмов шифрования и дешифрования шифровкой. Ключ — набор значений (чисел), которыми оперируют алгоритмы шифрования и дешифрования.

Обратите внимание, что шифрование симметричными ключами использует единственный ключ (ключ, содержащий непосредственно набор кодируемых значений) и для кодирования и для дешифрования. Кроме того, алгоритмы шифрования и дешифрования — инверсии друг друга. Если — обычный текст, — зашифрованный текст, а — ключ, алгоритм кодирования создает зашифрованный текст из исходного текста.

Алгоритм же дешифрования Dk (x) создает исходный текст из зашифрованного текста. Мы предполагаем, что и обратны друг другу. Они применяются, последовательно преобразуя информацию из одного вида в другой и обратно.

# 3 Выполнение лабораторной работы [1]

## 3.1 Шифр 1

Маршрутное шифрование включает в себя несколько преобразований изначального текста для корректной шифровки и расшифровки. Из-за некоторых особенностей встроенной функции reshape(array, dims...) в Julia некоторые функции приходилось дополнительно прописывать транспонирование матрицы получавшихся символов текста.

function route\_encrypt(text::AbstractString, rows::Int)

# Заполним текст пробелами, чтобы длина была кратна rows

len = length(text)

cols = ceil(Int, len / rows)

padded\_len = rows \* cols

padded\_text = rpad(text, padded\_len, ' ')

# Переводим строку в массив символов

arr = collect(padded\_text)

# Формируем матрицу rows x cols

mat = reshape(arr, cols, rows)' # Транспонируем после reshape

# Считываем по колонкам (столбцам)

encrypted = join(mat[:])

return encrypted

end

route\_encrypt (generic function with 1 method)

julia>

julia> function route\_decrypt(cipher::AbstractString, rows::Int)

len = length(cipher)

cols = len ÷ rows

arr = collect(cipher)

# Формируем матрицу rows x cols

mat = reshape(arr, rows, cols)

# Транспонируем для восстановления исходной матрицы

decrypted\_mat = mat'

decrypted = join(decrypted\_mat[:])

return strip(decrypted) # Убираем добавленные пробелы

end

route\_decrypt (generic function with 1 method)

julia>

julia> # Пример использования

julia> text = "Пример маршрутного шифрования"

"Пример маршрутного шифрования"

julia> rows = 5

5

julia>

julia> encrypted = route\_encrypt(text, rows)

ERROR: MethodError: no method matching adjoint(::Char)

The function adjoint exists, but no method is defined for this combination of argument types.

Closest candidates are:

adjoint(::Missing)

@ Base missing.jl:101

adjoint(::LinearAlgebra.AdjointFactorization)

@ LinearAlgebra C:\Users\eliza\AppData\Local\Programs\Julia-1.11.7\share\julia\stdlib\v1.11\LinearAlgebra\src\factorization.jl:61

adjoint(::LinearAlgebra.LowerTriangular)

@ LinearAlgebra C:\Users\eliza\AppData\Local\Programs\Julia-1.11.7\share\julia\stdlib\v1.11\LinearAlgebra\src\triangular.jl:490

...

Stacktrace:

[1] getindex

@ C:\Users\eliza\AppData\Local\Programs\Julia-1.11.7\share\julia\stdlib\v1.11\LinearAlgebra\src\adjtrans.jl:334 [inlined]

[2] \_unsafe\_getindex\_rs

@ .\reshapedarray.jl:276 [inlined]

[3] \_unsafe\_getindex

@ .\reshapedarray.jl:273 [inlined]

[4] getindex

@ .\reshapedarray.jl:261 [inlined]

[5] macro expansion

@ .\multidimensional.jl:940 [inlined]

[6] macro expansion

@ .\cartesian.jl:64 [inlined]

[7] \_unsafe\_getindex!

@ .\multidimensional.jl:938 [inlined]

[8] \_unsafe\_getindex

@ .\multidimensional.jl:929 [inlined]

[9] \_getindex

@ .\multidimensional.jl:915 [inlined]

[10] getindex

@ .\abstractarray.jl:1312 [inlined]

[11] route\_encrypt(text::String, rows::Int64)

@ Main .\REPL[32]:15

[12] top-level scope

@ REPL[37]:1

julia> println("Зашифрованный текст: ", encrypted)

ERROR: UndefVarError: encrypted not defined in Main

Suggestion: check for spelling errors or missing imports.

Stacktrace:

[1] top-level scope

@ REPL[38]:1

julia>

julia> decrypted = route\_decrypt(encrypted, rows)

ERROR: UndefVarError: encrypted not defined in Main

Suggestion: check for spelling errors or missing imports.

Stacktrace:

[1] top-level scope

@ REPL[39]:1

julia> println("Расшифрованный текст: ", decrypted)

ERROR: UndefVarError: decrypted not defined in Main

Suggestion: check for spelling errors or missing imports.

Stacktrace:

[1] top-level scope

@ REPL[40]:1

julia>

julia> function route\_encrypt(text::AbstractString, rows::Int)

len = length(text)

cols = ceil(Int, len / rows)

padded\_len = rows \* cols

padded\_text = rpad(text, padded\_len, ' ')

arr = collect(padded\_text)

# reshape возвращает матрицу cols x rows,

# поэтому меняем на permutedims для транспонирования

mat = permutedims(reshape(arr, cols, rows))

При проверке правильности реализации важно учитывать, что шифры перестановки (а, значит, и маршрутное шифрование) относятся к симметричным шифрам. Это важно при проверке правильности работы шифра, для чего изначальное сообщение мы пропускаем через функции шифровки и расшифровки с одними и теми же параметрами (в частности, если параметры были изменены в функции шифровки для соответствия алгоритму, они выводились дополнительными переменными в результате выполнения функции). Так мы должны получить шифрокод после запуска функции шифровки, и изначальное сообщение после запуска функции расшифровки с теми же дополнительными параметрами на входе.

encrypted = join(mat[:])

return encrypted

end

route\_encrypt (generic function with 1 method)

julia>

julia> function route\_decrypt(cipher::AbstractString, rows::Int)

len = length(cipher)

cols = len ÷ rows

arr = collect(cipher)

mat = reshape(arr, rows, cols)

decrypted\_mat = permutedims(mat)

decrypted = join(decrypted\_mat[:])

return strip(decrypted)

end

route\_decrypt (generic function with 1 method)

julia>

julia> text = "Пример маршрутного шифрования"

"Пример маршрутного шифрования"

julia> rows = 5

5

julia>

julia> encrypted = route\_encrypt(text, rows)

"П у врмтшаианинмрофиешгрярроо "

julia> println("Зашифрованный текст: ", encrypted)

Зашифрованный текст: П у врмтшаианинмрофиешгрярроо

julia>

julia> decrypted = route\_decrypt(encrypted, rows)

"Пример маршрутного шифрования"

julia> println("Расшифрованный текст: ", decrypted)

Расшифрованный текст: Пример маршрутного шифрования

julia>Результат работы кода представлен ниже (рис. 1).

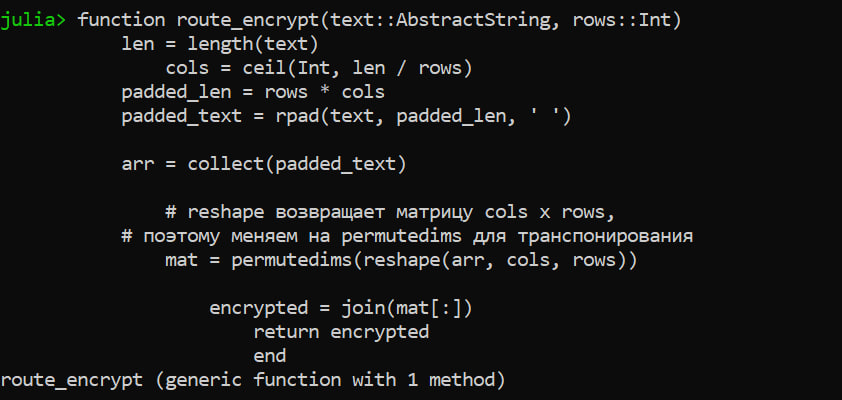


Рис. 1.1: Результат работы маршрутного шифрования

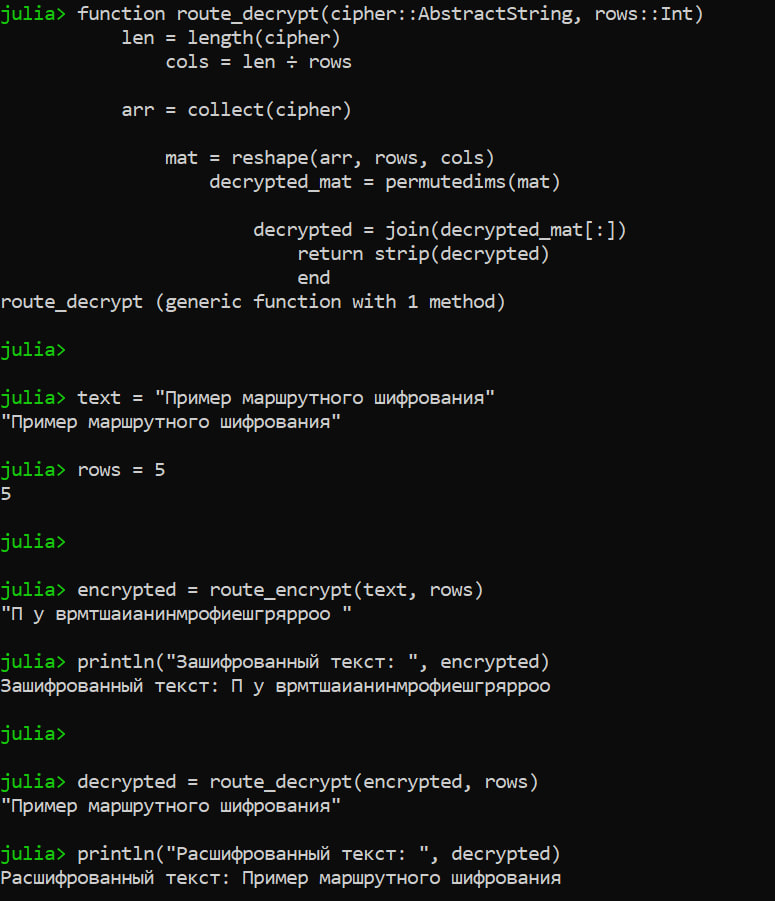


Рис. 1.2: Результат работы маршрутного шифрования

## 3.2 Шифрование с помощью решёток

Для реализации шифрования с помощью решёток использовались множество функций для работы с массивами, такие как findfirst(x::function, array), rotr90(A[, k]) и rotl90(A[, k]), классический конструктор массива Array{Type, N\_of\_dims}(undef, dims...) и прочие [3].

При проверке правильности реализации важно учитывать, что шифры перестановки (а, значит, и шифрование с помощью решёток) относятся к симметричным шифрам. Это важно при проверке правильности работы шифра, для чего изначальное сообщение мы пропускаем через функции шифровки и расшифровки с одними и теми же параметрами (в частности, если параметры были изменены в функции шифровки для соответствия алгоритму, они выводились дополнительными переменными в результате выполнения функции). Так мы должны получить шифрокод после запуска функции шифровки, и изначальное сообщение после запуска функции расшифровки с теми же дополнительными параметрами на входе.

Результат работы кода представлен ниже (рис. 2).

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 2.1: Результат работы шифрования с помощью решёток

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 2.2: Результат работы шифрования с помощью решёток

## 3.3 Таблицы Виженера

Для реализации таблицы Виженера необходимо было ограничить алфавит. В тексте лабораторной работы [1] предложен пример использования исключительно латиницы. В своей реализации я предлагаю использовать в качестве алфавита все символы ASCII, которые доступны в Julia [3].

В языке Julia число ASCII символов ограничено [4], которые и были алфавитом в использованной реализации шифрования с помощью таблиц Виженера.

Елизавета Савченко, [25.09.2025 11:54]

julia> function rotate\_left90(A::Array{Bool,2}, k::Int=1)

for \_ in 1:k

A = permutedims(A, (2,1))[:, end:-1:1]

end

return A

end

rotate\_left90 (generic function with 2 methods)

julia>

julia> function grille\_encrypt(text::AbstractString, grille::Array{Bool,2})

n, m = size(grille)

total\_cells = n \* m

chars = collect(text) # массив символов

# Если текста меньше, дополним пробелами

if length(chars) < total\_cells

append!(chars, [' ' for \_ in 1:(total\_cells - length(chars))])

else

chars = chars[1:total\_cells]

end

mat = Array{Char}(undef, n, m)

fill!(mat, ' ')

idx = 1

for rot in 0:3

current\_mask = rotate\_left90(grille, rot) # определите функцию поворота

for i in 1:n, j in 1:m

if current\_mask[i,j] && mat[i,j] == ' '

mat[i,j] = chars[idx]

idx += 1

end

end

end

return join(vec(mat))

end

grille\_encrypt (generic function with 1 method)

julia> grille = [

true false false false;

false false true false;

false true false false;

false false false true

]

4×4 Matrix{Bool}:

1 0 0 0

0 0 1 0

0 1 0 0

0 0 0 1

julia>

julia> text = "Секретноешифрование123"

"Секретноешифрование123"

julia> encrypted = grille\_encrypt(text, grille)

"С о тк ен е р"

julia> println("Зашифрованный текст:")

Зашифрованный текст:

julia> println(encrypted)

Елизавета Савченко, [25.09.2025 12:11]

const ASCII\_SIZE = 128

128

julia> alphabet = [Char(i) for i in 0:ASCII\_SIZE-1]

128-element Vector{Char}:

'\0': ASCII/Unicode U+0000 (category Cc: Other, control)

'\x01': ASCII/Unicode U+0001 (category Cc: Other, control)

'\x02': ASCII/Unicode U+0002 (category Cc: Other, control)

'\x03': ASCII/Unicode U+0003 (category Cc: Other, control)

'\x04': ASCII/Unicode U+0004 (category Cc: Other, control)

'\x05': ASCII/Unicode U+0005 (category Cc: Other, control)

'\x06': ASCII/Unicode U+0006 (category Cc: Other, control)

'\a': ASCII/Unicode U+0007 (category Cc: Other, control)

'\b': ASCII/Unicode U+0008 (category Cc: Other, control)

'\t': ASCII/Unicode U+0009 (category Cc: Other, control)

'\n': ASCII/Unicode U+000A (category Cc: Other, control)

'\v': ASCII/Unicode U+000B (category Cc: Other, control)

'\f': ASCII/Unicode U+000C (category Cc: Other, control)

⋮

't': ASCII/Unicode U+0074 (category Ll: Letter, lowercase)

'u': ASCII/Unicode U+0075 (category Ll: Letter, lowercase)

'v': ASCII/Unicode U+0076 (category Ll: Letter, lowercase)

'w': ASCII/Unicode U+0077 (category Ll: Letter, lowercase)

'x': ASCII/Unicode U+0078 (category Ll: Letter, lowercase)

'y': ASCII/Unicode U+0079 (category Ll: Letter, lowercase)

'z': ASCII/Unicode U+007A (category Ll: Letter, lowercase)

'{': ASCII/Unicode U+007B (category Ps: Punctuation, open)

'|': ASCII/Unicode U+007C (category Sm: Symbol, math)

'}': ASCII/Unicode U+007D (category Pe: Punctuation, close)

'~': ASCII/Unicode U+007E (category Sm: Symbol, math)

'\x7f': ASCII/Unicode U+007F (category Cc: Other, control)

julia> vigenere\_table = Array{Char}(undef, ASCII\_SIZE, ASCII\_SIZE)

128×128 Matrix{Char}:

'\x00\x00\x00\x10' '\0' '\0' '\0' '\0' '\0' '\0' … '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x21\xe1\x3b\xd4' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x69\xc3\x72\x29' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x00\x00\x00\x14' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' … '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\xb9\x3a\x5e\x60' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x00\x00\x01\xcb' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\xff\xf0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\xff\xff\xff\xff' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x22\xcb\x01\xf7' '\0' '\0' '\0' '\0' '\0' '\0' … '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\xdd\xc7\xae\xb2' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\x43\xc5\xf5\xc5' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

⋮ ⋮ ⋱ ⋮ ⋮

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' … '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

Елизавета Савченко, [25.09.2025 12:11]

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' … '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'

julia>

julia> for i in 0:ASCII\_SIZE-1

for j in 0:ASCII\_SIZE-1

vigenere\_table[i+1, j+1] = Char(mod(i + j, ASCII\_SIZE))

end

end

julia> row\_idx, col\_idx = 11, 21

(11, 21)

julia> println("Символ в таблице Виженера [строка=$row\_idx, столбец=$col\_idx]: ", vigenere\_table[row\_idx, col\_idx])

Символ в таблице Виженера [строка=11, столбец=21]: ▲

julia>

julia> # Вывести всю 11-ю строку

julia> println("11-я строка таблицы Виженера:")

11-я строка таблицы Виженера:

julia> println(join(vigenere\_table[11, :]))

♫☼►◄↕️‼️¶§▬↨↑↓→∟↔️▲▼123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~⌂�☺️☻♥️♦️♣️♠️При проверке правильности реализации важно учитывать, что шифры перестановки (а, значит, и шифрование с помощью таблицы Виженера) относятся к симметричным шифрам. Это важно при проверке правильности работы шифра, для чего изначальное сообщение мы пропускаем через функции шифровки и расшифровки с одними и теми же параметрами (в частности, если параметры были изменены в функции шифровки для соответствия алгоритму, они выводились дополнительными переменными в результате выполнения функции). Так мы должны получить шифрокод после запуска функции шифровки, и изначальное сообщение после запуска функции расшифровки с теми же дополнительными параметрами на входе.

Результат работы кода представлен ниже (рис. 3).

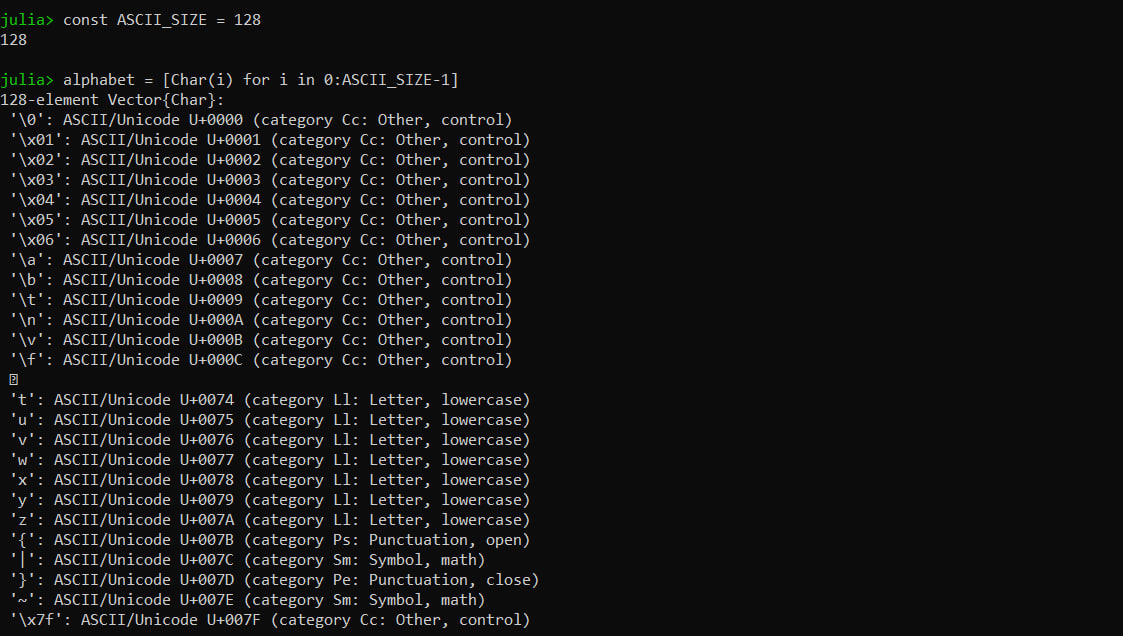


Рис. 3.1: Результат работы шифра с помощью таблиц Виженера

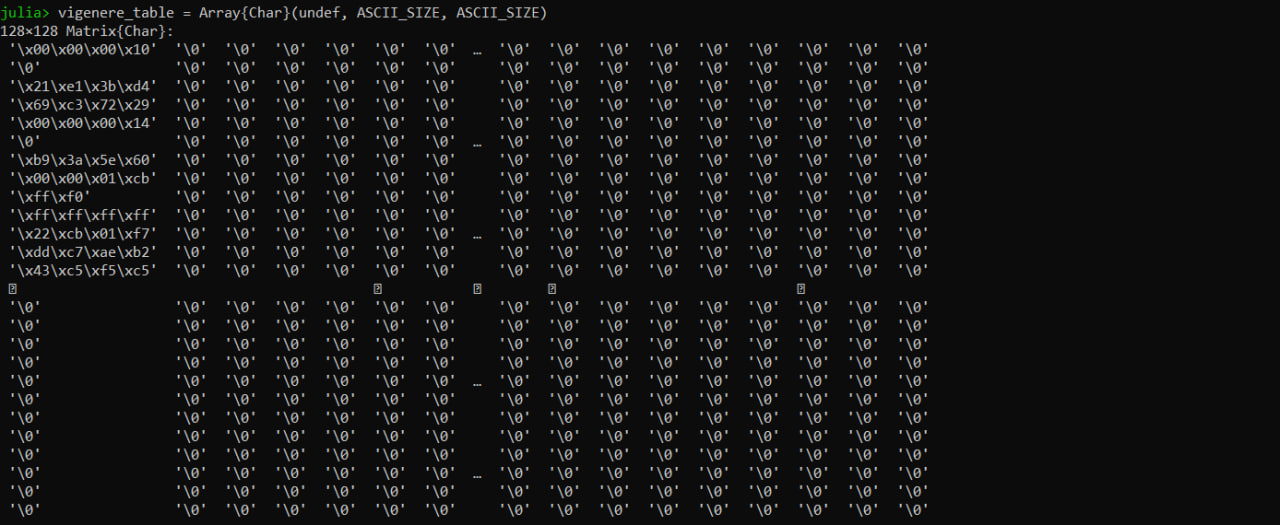


Рис. 3.2: Результат работы шифра с помощью таблиц Виженера

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 3.3: Результат работы шифра с помощью таблиц Виженера

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис. 3.4: Результат работы шифра с помощью таблиц Виженера

# 4 Выводы

В результате работы мы ознакомились с традиционными моноалфавитными шрифтами простой замены, а именно:

* Маршрутным шифрованием;
* Шифрованием с помощью решёток;
* Таблицами Виженера.

Также были записаны скринкасты:

На RuTube:

* [Весь плейлист](https://rutube.ru/plst/540770)
* [Выполнения лабораторной работы, часть 1](https://rutube.ru/video/10f37172e6c677ab27794fc40374c357)
* [Выполнения лабораторной работы, часть 2](https://rutube.ru/video/54382b0d694e12e25aa08a57c8801893)
* [Запись создания отчёта](https://rutube.ru/video/2620c9656b8ee7c3ab1c8784813b1904)
* [Запись создания презентации](https://rutube.ru/video/a67c8780c4ad31ebd371f5dc9188fdd0)
* [Защита лабораторной работы](https://rutube.ru/video/c0858c0cde854661e4cf9a98a3267598)

На Платформе:

* [Весь плейлист](https://plvideo.ru/playlist?list=vaNN02mO97J6)
* [Выполнения лабораторной работы, часть 1](https://plvideo.ru/watch?v=4Ces3XmW5SsX)
* [Выполнения лабораторной работы, часть 2](https://plvideo.ru/watch?v=zQvK3sfwqJA8)
* [Запись создания отчёта](https://plvideo.ru/watch?v=6LSqPhPtwkj5)
* [Запись создания презентации](https://plvideo.ru/watch?v=y2gwvx0xzRVl)
* [Защита лабораторной работы](https://plvideo.ru/watch?v=mpV0o8O6WOym)

# Список литературы

1. Лабораторная работа №2. Шифры перестановки [Электронный ресурс]. RUDN, 2024. URL: <https://esystem.rudn.ru/pluginfile.php/2368506/mod_folder/content/0/lab01.pdf>.

2. Математика криптографии и теория шифрования [Электронный ресурс]. URL: <https://intuit.ru/studies/courses/552/408/info>.

3. Julia 1.10 Documentation [Электронный ресурс]. 2024. URL: <https://docs.julialang.org/en/v1/>.

4. Julia 1.10 Documentation [Электронный ресурс]. 2024. URL: <https://docs.julialang.org/en/v1/base/strings/>.