Цель работы

Реализовать шифры: маршрутное шифрование, шифрование с помощью решеток и таблица Виженера

Выполнение лабораторной работы

Маршрутное шифрование

```
function route_encrypt(message, key, rows, cols)
        message =  filter(!isspace, message)
        matrix = fill('_', rows, cols)
        index = 1
        new_message = ""
        for i = 1:rows
                for j = 1:cols
                        if index != rows * cols
                                matrix[i, j] = message[index]
                                index += 1
                        end
                end
        end
        for j in sort(collect(key))
                for i = 1:rows
                        new_message *= (matrix[i, (findfirst(j, key))])
                end
        end
        return new_message
end

message = "this is beautiful life"
rows = 4
cols = 5
key = "water"
println(route_encrypt(message, key, rows, cols))
```

Шифрование с помощью решеток

```
function rails_encrypt(text, key, k)
        grid = fill(" ", 2 * k, 2 * k)
        matrix = fill(" ", k, k)
        index = 1
        new_message = ""
        text = replace(text, " " => "")
        for i in 1:k
                for j in 1:k
                        grid[i, j] = string(index)
                        matrix[i, j] = string(index)
                        index +=1
                end
        end
        for i = 1:(size(grid)[1])
                for j = (size(grid)[1]):-1:1
                        if grid[i, j] == " "
                                matrix = rotr90(matrix)
                                grid[(i+k-1):-1:i, j:-1(j-k+1)] = matrix[k:-1:1, k:-1:1]
                        end
                end
        end
index = 1
arr = Vector{String}()

for r in text
        checker = false
        for i = 1:(size(grid)[2])
                for j = 1:(size(grid)[2])
                        if grid[i,j] == string(index) && checker == false
                                if ((string())))
```

```
                        grid[i, j] = string(r)
                        push!(arr, string(i, " ", j))
                        checker = true
                    end
                end

                if checker
                    break
                end
            end
            if checker
                break
            end
        end

        if checker
            index += 1
            if index > k^2
                index = 1
                empty!(arr)
            end
        end
    end

    # читаем по ключу (столбцами)
```

```
for ch in sort(collect(key))
    col = findfirst(==(ch), key)
    if col === nothing
        continue
    end

    for i in 1:2k
        new_message *= grid[i, col]

        # если последний символ оказался цифрой (старые индексы), за
        if isdigit(new_message[end])
            new_message = new_message[1:end-1] * " "
        end
    end
```

Таблица Виженера

```
function vigenere_encrypt(text,key)
        alphabet = 'a':'z'
        output = ""
        key_index = 1


        for i in text
                if isletter(i)
                        offset = findfirst(isequal(key[key_index]), alphabet) - 1
                        index = findfirst(isequal(i), alphabet) + offset
                        index > 26 && (index -= 26)
                        output *=alphabet[index]
                        key_index += 1
                        key_index > length(key) && (key_index = 1)
                else
                        output *= i
                end
        end
        return output
end

text = "hello crazy world"
key = "cake"
println(vigenere_encrypt(text, key))
```

```
    end
    return codes
end

function codes_to_string(codes::Vector{Int})
    str = ""
    for code in codes
        if haskey(num_to_char, code)
            str *= num_to_char[code]
        else
            error("Код $code не найден в алфавите.")
        end
    end
    return str
end

function encrypt(message::String, key::String)
    msg_codes = string_to_codes(message)
    key_codes = string_to_codes(key)


    n = length(msg_codes)
    if n == 0
        return ""
    end

    extended_key = repeat(key_codes, ceil(Int, n / length(key_codes)))[1:n]

    cipher_codes = Int[]
    for i in 1:n
```

```
        sum_val = msg_codes[i] + extended_key[i]
        code = mod(sum_val, 26)
        if code == 0
            code = 26
        end
        push!(cipher_codes, code)
    end
```

Вывод

В ходе выполнения лабораторной работы были реализованы три метода шифрования