Лабораторная работа 3

Шифрование гаммированием конечной гаммы

Климин Никита Денисович

Содержание

# 1. Цель работы

Реализовать алгоритм шифрования гаммированием конечной гаммы на языке julia. Проверить работу программы на примере расшифровки и зашифровки текста.

# 2. Задание

Реализовать алгоритм шифрования гаммированием конечной гаммы на языке julia. Проверить работу программы на примере расшифровки и зашифровки текста.

# 3. Теоретическое введение

**Гаммирование** — это метод симметричного шифрования текста, при котором каждая буква исходного текста преобразуется с помощью ключа (гаммы) по модулю длины алфавита.

* Простейший вариант — **одноразовое использование ключа**, где длина ключа равна длине текста.
* Шифрование выполняется операцией сложения по модулю 2 (XOR) для двоичных данных или по модулю N для букв (N — длина алфавита).
* Дешифрование осуществляется повторным применением той же операции.

Формулы для шифрования и расшифровки букв русского алфавита (N = 33): C\_i = (P\_i + K\_i) mod 33 # шифрование P\_i = (C\_i - K\_i) mod 33 # расшифровка

* P\_i — порядковый номер буквы исходного текста
* K\_i — порядковый номер буквы ключа
* C\_i — порядковый номер зашифрованной буквы

# 4. Выполнение лабораторной работы

Программа была написана на Julia.

alphabet = ['А':'Я'...] #создаём массив русского алфавита  
num(ch) = findfirst(==(ch), alphabet) # функция перевода букв в номер  
lettr(n) = alphabet[n] # функция перевода номера обратно в букву  
  
function gamma(text::String, key::String; decrypt=false) # функция гаммирования. text - исходный текст, key - гамма  
 t = uppercase(text) # переводим текст в верхний регистр  
 k = replace(uppercase(key), " " => "") # переводим гамму в верхний регистр и удаляем пробелы  
 tn = [ch for ch in t] # преобразуем текст в масив символов  
 kn = [num.(collect(k))...] # преобразуем ключ в масив номеров букв  
 kn = repeat(kn, ceil(Int, length(tn)/length(kn)))[1:length(tn)] # если ключ короче текста, то повторяем его до нужной длинны  
 op = decrypt ? (-) : (+) # определяем операцию шифрования или расшифрования  
  
 r = [ # основный цикл шифрования и расшифрования  
 ch == ' ' ? ' ' : lettr(mod1(op(num(ch) ,y), length(alphabet))) # если символ пробел, то оставляем его без изменений.   
 for (ch, y) in zip(tn, kn) # иначе берём номер буквы и применяем сложение/вычитание по модулю и переводим обратно в букву  
 ]  
 return join(r)  
end  
  
println("введите текст")  
text = readline()  
println("введите гамму")  
key = readline()  
println("выберите действие, 1- шифрование, 2 - дешифрование")  
choice = readline()  
  
if choice == "1"  
 result = gamma(text, key)  
 println("зашифрованный текст $result")  
elseif choice == "2"  
 result = gamma(text, key, decrypt=true)  
 println("расшифрованный текст $result")  
else  
 println("введите 1 или 2")  
end

**Пример работы программы в терминале**

|  |
| --- |
| Рисунок 1: Пример работы программы |

# 5. Выводы

Реализован алгоритм шифрования гаммированием конечной гаммой. Программы коректно шифрует и расшифровывает текст на примере слова ПРИКАЗ

# Список литературы