ЛАБОРАТОРНАЯ РАБОТА №4

Вычисление наибольшего общего делителя

Кузнецов Юрий Владимирович

# Введение

В данном отчёте будет представлена реализация вычисления наибольшего общего делителя

# Основное содержание

## Шифры простой замены:

* Алгоритм Евклида
* Бинарный алгоритм Евклида
* Расширенный алгоритм Евклида
* Расширенный бинарный алгоритм Евклида

# Кодовая реализация

## Алгоритм Евклида

function euclid(a::Int, b::Int)::Int  
 while a != 0 && b != 0  
 if a >= b  
 a %= b  
 else  
 b %= a  
 end  
 end  
 return a != 0 ? a : b  
end  
  
println(euclid(a, b))

## Бинарный алгоритм Евклида

function euclid\_bin(a::Int, b::Int)::Int  
 g = 1  
 while a % 2 == 0 && b % 2 == 0  
 a ÷= 2  
 b ÷= 2  
 g \*= 2  
 end  
 u, v = a, b  
 while u != 0  
 if u % 2 == 0  
 u ÷= 2  
 elseif v % 2 == 0  
 v ÷= 2  
 elseif u >= v  
 u -= v  
 else  
 v -= u  
 end  
 end  
 return g \* v  
end  
  
println(euclid\_bin(a, b))

## Расширенный алгоритм Евклида

function euclid\_ext(a::Int, b::Int)::Tuple{Int,Int,Int}  
 if a == 0  
 return b, 0, 1  
 else  
 div, x, y = euclid\_ext(b % a, a)  
 return div, y - (b ÷ a) \* x, x  
 end  
end  
  
println(euclid\_ext(a, b))

## Шифрование с помощью решеток

function euclid\_bin\_ext(a::Int, b::Int)::Tuple{Int,Int,Int}  
 g = 1  
 while a % 2 == 0 && b % 2 == 0  
 a ÷= 2  
 b ÷= 2  
 g \*= 2  
 end  
 u, v = a, b  
 A, B, C, D = 1, 0, 0, 1  
 while u != 0  
 if u % 2 == 0  
 u ÷= 2  
 if A % 2 == 0 && B % 2 == 0  
 A ÷= 2  
 B ÷= 2  
 else  
 A = (A + b) ÷ 2  
 B = (B - a) ÷ 2  
 end  
 elseif v % 2 == 0  
 v ÷= 2  
 if C % 2 == 0 && D % 2 == 0  
 C ÷= 2  
 D ÷= 2  
 else  
 C = (C + b) ÷ 2  
 D = (D - a) ÷ 2  
 end  
 elseif u >= v  
 u -= v  
 A -= C  
 B -= D  
 else  
 v -= u  
 C -= A  
 D -= B  
 end  
 end  
 return g \* v, C, D  
end  
  
println(euclid\_bin\_ext(a, b))

## Расширенный бинарный алгоритм Евклида

char\_sequence = 1  
 used\_positions = String[]  
 for char in sanitized\_text  
 placed = false  
 for i in 1:(2\*matrix\_dim), j in 1:(2\*matrix\_dim)  
 if encryption\_matrix[i, j] == string(char\_sequence) && !placed  
 position\_key = string(i, ",", j)  
 if !(position\_key in used\_positions)  
 encryption\_matrix[i, j] = string(char)  
 push!(used\_positions, position\_key)  
 placed = true  
 end  
 end  
 end  
 char\_sequence += 1  
 if char\_sequence > matrix\_dim^2  
 char\_sequence = 1  
 empty!(used\_positions)  
 end  
 end

# Заключение

В данной лабораторной работе были реализованы вычисления наибольшего общего делителя