# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

### Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3 =============== ## Тема: Шифрование гаммированием дисциплина: Математические основы защиты информации и информационной безопасности

Студент: Койфман Кирилл Дмитриевич

Группа: НФИмд-01-25

## Введение

### Цель работы

Получение практических навыков реализации алгоритмов, использующих гаммирование.

### Задачи

1. Реализовать алгоритм шифрования гаммированием конечной гаммой

## Теория

В основе функционирования шифров простой замены лежит следующий принцип: формируется -разрядная случайная двоичная последовательность — ключ шифра. Отправитель производит побитовое сложение по модулю два ( 2) ключа

и -разрядной двоичной последовательности

соответствующей посылаемому сообщению:

где -й бит исходного текста, -й бит ключа, операция побитового сложения (XOR), -й бит получившейся криптограммы

Операция побитного сложения является обратимой, т.е. $(х\oplusу)\oplusу = х$, поэтому дешифрование осуществляется повторным применением операции к криптограмме:

Однако основным недостатком такой схемы является равенство объёма ключевой информации и суммарного объёма передаваемых сообщений. Данный недостаток можно убрать, использовав ключ в качестве “зародыша”, порождающего значительно более длинную ключевую последовательность.

*Гаммированием* называют процедуру наложения при помощи некоторой функции на исходный текст гаммы шифра, т.е. псевдослучайной последовательности (ПСП) с выходов генератора . Псевдослучайная последовательность по своим статистическим свойствам неотличима от случайной последовательности, но является детерминированной, т.е. известен алгоритм ее формирования. Обычно в качестве функции берется операция поразрядного сложения по модулю два или по модулю ( число букв алфавита открытого текста).

Простейший генератор псевдослучайной последовательности можно представить рекуррентным соотношением:

где -й член последовательности псевдослучайных чисел, — ключевые параметры. Такая последовательность состоит из целых чисел от 0 до . Если элементы и совпадут, то совпадут и последующие участки: , . Таким образом, ПСП является периодической. Знание периода гаммы cущественно облегчает криптоанализ. Максимальная длина периода равна . Для ее достижения необходимо удовлетворить следующим условиям: 1. и — взаимно простые числа; 2. делится на любой простой делитель числа ; 3. кратно 4, если кратно 4.

Стойкость шифров, основанных на процедуре гаммирования, зависит от характеристик гаммы — длины и равномерности распределения вероятностей появления знаков гаммы.

При использовании генератора ПСП получаем бесконечную гамму. Однако, возможен режим шифрования конечной гаммы. В роли конечной гаммы может выступать фраза. Как и ранее, используется алфавитный порядок букв, т.е. буква “а” имеет порядковый номер 1, “б” - 2 и т.д.

## ## Ход работы

Для решения поставленной задачи реализуем алгоритм шифрования гаммированием конечной гаммой на языке программирования C++ (Листинг-1), а также проведём тест данного алгоритма, чтобы проверить корректность его работы (Листинг-2):

#include <iostream>  
#include <cstdint>  
#include <map>  
  
std::wostream& operator<<(std::wostream& out, const std::map<wchar\_t, std::uint32\_t>& alphabet)  
{  
 out << "Alphabet:\n";  
 int i = 0;  
 for (const auto& pair : alphabet)  
 {  
 out << '[' << pair.first << ']' << '{' << pair.second << '}' << '\t';  
 if (++i % 10 == 0)  
 out << '\n';  
 }  
  
 return out;  
}  
  
void printTextCodes(const std::wstring& enteredText, const std::map<wchar\_t, std::uint32\_t>& alphabet)  
{  
 for (const auto& symbol : enteredText)  
 {  
 std::wcout << alphabet.at(symbol) << '\t';  
 }  
}  
  
int main()  
{  
 setlocale(LC\_ALL, "");  
  
 //Define an alphabet with use of map  
 const wchar\_t alphabetBeginSymbol = L'А';  
 const wchar\_t alphabetEndSymbol = L'Я';  
 const int alphabethLength = alphabetEndSymbol - alphabetBeginSymbol + 1;  
 std::map<wchar\_t, std::uint32\_t> alphabet{};  
 for (std::uint32\_t symbol\_index = 1; symbol\_index <= alphabethLength; ++symbol\_index)  
 {  
 alphabet.insert({ alphabetBeginSymbol + (symbol\_index - 1), symbol\_index });  
 }  
 std::wcout << alphabet << '\n';  
 std::wcout << "Alphabet length: " << alphabethLength << '\n';  
  
 //Define input open message  
 std::wstring enteredMessage = L"ПРИКАЗ";  
 size\_t enteredMessageLength = enteredMessage.size();  
 //std::wcin >> enteredMessage;  
 std::wcout << "Entered message: " << enteredMessage << '\n';  
 std::wcout << "Entered message(codes):\n";  
 printTextCodes(enteredMessage, alphabet);  
 std::wcout << '\n';  
  
 //Define input gamma  
 std::wstring enteredGamma = L"ГАММА";  
 size\_t enteredGammaLength = enteredGamma.size();  
 //!std::wcin >> enteredGamma;  
 std::wcout << "Entered gamma: " << enteredGamma << '\n';  
 std::wcout << "Entered gamma(codes):\n";  
 printTextCodes(enteredGamma, alphabet);  
 std::wcout << '\n';  
  
 //Perform encrypting (by applying gamma)  
 std::wcout << "-----------------------------------ENCRYPTING-----------------------------------\n";  
 std::wstring encryptedMessage;  
  
 for (std::uint32\_t symbol\_index = 0; symbol\_index < enteredMessageLength; ++symbol\_index)  
 {  
 //ENCRYPTED\_SYMBOL\_CODE = ENTERED\_MESSAGE\_CODE + ENTERED\_GAMMA % ALPHABET\_LENGTH  
 std::uint32\_t encryptedSymbolCode = (alphabet.at(enteredGamma[symbol\_index % enteredGammaLength]) +  
 (alphabet.at(enteredMessage[symbol\_index]) % alphabethLength)) % alphabethLength;  
  
 for (auto& element : alphabet)  
 if (element.second == encryptedSymbolCode)  
 {  
 encryptedMessage += element.first;  
 break;  
 }  
 }  
  
 std::wcout << "Encrypted message: " << encryptedMessage << '\n';  
 std::wcout << "Encrypted message(codes):\n";  
 printTextCodes(encryptedMessage, alphabet);  
 std::wcout << '\n';  
}

*Листинг-2(алгоритм, реализующий метод шифрования гаммирования конечной гаммой)*

[А]{1} [Б]{2} [В]{3} [Г]{4} [Д]{5} [Е]{6} [Ж]{7} [З]{8} [И]{9} [Й]{10}  
[К]{11} [Л]{12} [М]{13} [Н]{14} [О]{15} [П]{16} [Р]{17} [С]{18} [Т]{19} [У]{20}  
[Ф]{21} [Х]{22} [Ц]{23} [Ч]{24} [Ш]{25} [Щ]{26} [Ъ]{27} [Ы]{28} [Ь]{29} [Э]{30}  
[Ю]{31} [Я]{32}  
Alphabet length: 32  
Entered message: ПРИКАЗ  
Entered message(codes):  
16 17 9 11 1 8  
Entered gamma: ГАММА  
Entered gamma(codes):  
4 1 13 13 1  
-----------------------------------ENCRYPTING-----------------------------------  
Encrypted message: УСХЧБЛ  
Encrypted message(codes):  
20 18 22 24 2 12

*Листинг-2(результаты работы алгоритма шифрования гаммирования конечной гаммой)*

Исходя из полученных результатов (Листинг-2), можно судить о том, что данный алгоритм производит успешные шифрование вводимого текста, последовательно накладывая элементы гаммы шифра на каждый символ исходного текста.

## Заключение

В ходе проделанной лабораторной работы мной были усвоены знания по принципам работы с шифрами гаммирования, а также получены навыки по их реализации.