

Математические основы защиты информации и информационной безопасности.

Лабораторная работа №2.

Подмогильный Иван Александрович.

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	10

List of Figures

3.1	зашивровки кодов конечной гаммой	7
3.2	заголовочный файл	8
3.3	CMakeLists.txt файл	8
3.4	main.cpp файл	9
3.5	Результаты тестов	9

List of Tables

1 Цель работы

Освоить на практике шифрование гаммированием

2 Задание

1. Реализовать шифрование функцию шифрования гаммированием конечной гаммой

3 Выполнение лабораторной работы

Написал код для зашивровки кодов конечной гаммой. Тело функции:

```
//  
// Created by pi on 17.09.2022.  
//  
  
#include "../include/CipherFiniteGammaHelper.h"  
  
std::string CipherFiniteGammaHelper::engAlphabetLower = "abcdefghijklmnopqrstuvwxyz";  
std::string CipherFiniteGammaHelper::engAlphabetUpper = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
std::vector<std::string> CipherFiniteGammaHelper::rusAlphabetLower = {"a", "б", "в", "г", "д", "е", "ж", "з", "и", "й", "к",  
    "л", "м", "н", "о", "п", "р", "с", "т", "у", "ф", "х", "ц",  
    "ч", "ш", "щ", "ъ", "ы", "ь", "э", "ю", "я"};  
std::vector<std::string> CipherFiniteGammaHelper::rusAlphabetUpper = {"А", "Б", "В", "Г", "Д", "Е", "Ж", "З", "И", "Й", "К",  
    "Л", "М", "Н", "О", "П", "Р", "С", "Т", "У", "Ф", "Х", "Ц",  
    "Ч", "Ш", "Щ", "Ъ", "Ы", "Ь", "Э", "Ю", "Я"};  
  
template <typename T>  
void CipherFiniteGammaHelper::cipher(const T &msg, const T &pwd, T &encrypted){  
    int modulo = 26;  
    char msgIndexes[msg.size()], pwdIndexes[pwd.size()];  
    for (int i = 0; i < msg.size(); i++){  
        msgIndexes[i] = engAlphabetLower.find(msg[i]);  
    }  
    for (int i = 0; i < pwd.size(); i++){  
        pwdIndexes[i] = engAlphabetLower.find(pwd[i]);  
    }  
  
    for (int i = 0; i < msg.size(); i++){  
        int index = (msgIndexes[i] + pwdIndexes[i % pwd.size()]) % modulo;  
        encrypted[i] = engAlphabetLower[index];  
    }  
}  
  
template <typename T>  
void CipherFiniteGammaHelper::decipher(const T &enc, const T &pwd, T &decr) {  
    cipher(enc, pwd, &decr);  
}  
  
// so, this is as an easy fix, but if so, you need to list all the possible usages. Let's say that it's ok. By now  
// the program was tested with vectors and string. You can always switch back to the template definition inside .h file  
// then you don't need these lines.  
template void CipherFiniteGammaHelper::cipher(const std::string &msg, const std::string &pwd, std::string &encrypted);  
template void CipherFiniteGammaHelper::decipher(const std::string &enc, const std::string &pwd, std::string &decr);  
template void CipherFiniteGammaHelper::cipher(const std::vector<char> &msg, const std::vector<char> &pwd, std::vector<char> &encrypted);
```

Figure 3.1: зашивровки кодов конечной гаммой

Заголовочный файл:

```

#ifndef LAB03_CIPHERFINITEGAMMAHELPER_H
#define LAB03_CIPHERFINITEGAMMAHELPER_H

#include <string>
#include <vector>

class CipherFiniteGammaHelper{
public:
    // so, the user might want to insert the vector of chars, or the array of chars, or the string, or wstring.
    // so, let's use the template for these purposes
    template <typename T>
    static void cipher(const T& msg, const T& pwd, T& encrypted);
    // {
    //     int modulo = 26;
    //     char msgIndexes[msg.size()], pwdIndexes[pwd.size()];
    //     for (int i = 0; i < msg.size(); i++){
    //         msgIndexes[i] = engAlphabetLower.find(msg[i]);
    //     }
    //     for (int i = 0; i < pwd.size(); i++){
    //         pwdIndexes[i] = engAlphabetLower.find(pwd[i]);
    //     }
    //     for (int i = 0; i < msg.size(); i++){
    //         int index = (msgIndexes[i] + pwdIndexes[i % pwd.size()]) % modulo;
    //         encrypted[i] = engAlphabetLower[index];
    //     }
    // };

    template <typename T>
    static void decipher(const T& enc, const T& pwd, T& decr);
    // {
    //     cipher(enc, pwd, decr);
    // };

    static std::string engAlphabetLower;
    static std::string engAlphabetUpper;
    static std::vector<std::string> rusAlphabetLower;
    static std::vector<std::string> rusAlphabetUpper;

    // static void find_in_rus()
    // };
};

```

Figure 3.2: заголовочный файл

Написал CMakeLists.txt файл.

```

cmake_minimum_required(VERSION 3.20)
project(lab03)

set(CMAKE_CXX_STANDARD 14)

add_library(lab03Lib include/CipherFiniteGammaHelper.h src/CipherFiniteGammaHelper.cpp)

add_executable(main src/main.cpp)
target_link_libraries(main lab03Lib)

```

Figure 3.3: CMakeLists.txt файл

Написал main.cpp файл, в котором есть тесты реализованных функций.


```

1  #include <iostream>
2  #include "../include/CipherFiniteGammaHelper.h"
3
4  int main() {
5      std::string test1 = "hello, world!";
6      std::string enc1( test1.size(), ' ');
7      std::string dec1( test1.size(), ' ');
8      std::string pwd1( 5, "parol");
9
10     std::vector<char> test2 = {'h', 'e', 'l', 'l', 'o'};
11     std::vector<char> enc2(test2.size(), ' ');
12     std::vector<char> dec2(test2.size(), ' ');
13     std::vector<char> pwd2 = {'p', 'a', 'r', 'o', 'l'};
14
15     CipherFiniteGammaHelper::cipher(test1, pwd1, &enc1);
16     std::cout << enc1 << std::endl;
17
18     CipherFiniteGammaHelper::cipher(test2, pwd2, &enc2);
19     for (int i = 0; i < enc2.size(); i++){
20         std::cout << enc2[i];
21     }
22     std::cout << std::endl;
23
24     // CipherFiniteGammaHelper::decipher(enc1, pwd1, dec1);
25     // std::cout << dec1 << std::endl;
26
27
28     return 0;
29 }

```

Figure 3.4: main.cpp файл

Результаты тестов.

```

/home/pi/education/pfur_masters/mat0snovyInfBez/labs/lab03/cmake-build-debug/main
weczzrancads
weczz

Process finished with exit code 0

```

Figure 3.5: Результаты тестов

4 Выводы

Освоил на практике применения метода шифрования гаммированием с конечной гаммой