Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЁТ
к лабораторной работе №2
на тему

ЭЛЕМЕНТЫ КРИПТОГРАФИИ

Выполнил: студент гр. 253505
Сенько Н. С.

Проверил: ассистент кафедры информатики
Герчик А. В.

Минск 2025

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

Необходимо изучить и программно реализовать классические методы шифрования текстовой информации на примере шифра Цезаря и шифра Виженера. В ходе выполнения работы предполагается разработка программных средств, позволяющих выполнять процесс шифрования и дешифрования текстовых файлов с использованием данных методов. В ходе работы необходимо реализовать алгоритмы шифрования и дешифрования для обоих методов, а также разработать программный интерфейс для удобного взаимодействия пользователя с программой. Важно обеспечить возможность работы с текстовыми файлами, позволяя шифровать их содержимое и впоследствии расшифровывать его при наличии ключа.

2 ХОД РАБОТЫ

Шифр Цезаря – это метод шифрования, при котором каждая буква в открытом тексте заменяется другой буквой, находящейся на фиксированное число позиций дальше в алфавите. Например, при сдвиге на 5 буква «А» превращается в «Е».

Шифр Виженера – это метод многозначного шифрования, при котором используются различные сдвиги символов в зависимости от ключевого слова. Этот метод является улучшенной версией шифра Цезаря и позволяет значительно усложнить расшифровку без знания ключа. В ходе выполнения лабораторной работы были реализованы функции шифрования и дешифрования текста с использованием шифра Цезаря. Реализованы функции шифрования и дешифрования текста с использованием шифра Виженера. Разработан интерфейс для работы с текстовыми файлами, позволяющий пользователю выбрать метод шифрования, ввести ключ и зашифровать/расшифровать файл.

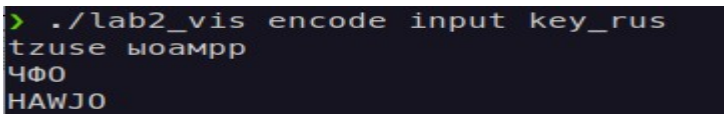

Рисунок 1 – Результат работы программы при шифровании

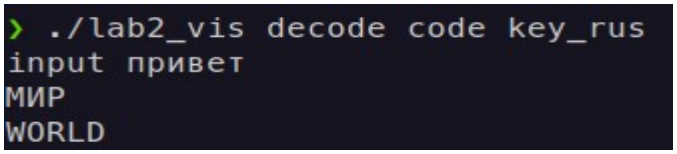

Рисунок 2 – Результат работы программы при дешифрации

Таким образом, в ходе выполнения лабораторной работы было реализовано и протестировано программное средство шифрования и дешифрования текстовых файлов двумя шифрами. В процессе разработки были изучены принципы работы выбранных алгоритмов шифрования, их особенности, преимущества и недостатки.

Программа была протестирована на различных входных данных, что позволило убедиться в её корректности и работоспособности. Кроме того, в ходе выполнения лабораторной работы была проведена отладка кода, исправлены возможные ошибки и оптимизированы некоторые участки программы для повышения её производительности.

ЗАКЛЮЧЕНИЕ

В ходе работы было реализовано и протестировано программное средство шифрования и дешифрования текстовых файлов шифрами Цезаря и

Вижинера. Были изучены принципы работы данных методов, их особенности и различия в устойчивости к криптоанализу.

Шифр Цезаря показал свою простоту в реализации, но низкую криптостойкость, что делает его пригодным лишь для базовых демонстрационных целей. В отличие от него, шифр Виженера обеспечивает более сложное шифрование за счёт использования ключевого слова, что значительно усложняет его взлом методом частотного анализа.

Разработанная программы позволяют пользователю выбрать метод шифрования, вводить ключ и получать зашифрованный или расшифрованный текст. В ходе тестирования подтвердилось корректное выполнение операций, а также была обеспечена возможность работы с текстовыми файлами. Таким образом, проделанная работа способствовала углублению знаний в области симметричного шифрования и алгоритмов защиты информации, а также позволила применить на практике навыки программирования для реализации криптографических методов.

Приложение А (обязательное)
Исходный код программного продукта

```
#include <iostream>
#include <fstream>
#include <string>
#include <locale>
#include <codecvt>

const std::wstring russian_lower = L"абвгдеёжзийклмнопрстувхцчшщьыъэюя";
const std::wstring russian_upper = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";
const std::wstring english_lower = L"abcdefghijklmnopqrstuvwxyz";
const std::wstring english_upper = L"ABCDEFGHIJKLMNOPQRSTUVWXYZ";

bool is_rus(wchar_t c) {
    return (c >= L'a' && c <= L'я') || (c >= L'A' && c <= L'Я');
}

bool is_en(wchar_t c) {
    return (c >= L'a' && c <= L'z') || (c >= L'A' && c <= L'Z');
}

bool is_lower_rus(wchar_t c) {
    return (c >= L'a' && c <= L'я');
}

bool is_lower_en(wchar_t c) {
    return (c >= L'a' && c <= L'z');
}

wchar_t caesar_shift(wchar_t c, int shift, const std::wstring& alphabet)
{
    size_t pos = alphabet.find(c);
    if (pos == std::wstring::npos) {
        return c; // Return character unchanged if not found in alphabet
    }
    size_t new_pos = (pos + shift + alphabet.size()) % alphabet.size();
    return alphabet[new_pos];
}

int main(int argc, char* argv[]) {
    setlocale(LC_CTYPE, "en_US.utf8");

    if (argc < 5) {
        std::cerr << "Usage: " << argv[0] << " <encode/decode>
<filename> <shift> <alphabet_size>\n";
        return 1;
    }

    std::string mode = argv[1];
    std::string filename = argv[2];
    int shift = std::stoi(argv[3]);
    int alphabet_size = std::stoi(argv[4]);

    // Open file with wide character support
    std::wifstream file(filename);
    file.imbue(std::locale(file.getloc(), new
std::codecvt_utf8<wchar_t>()));
```

```
        if (!file.is_open()) {
            std::cerr << "Error opening file: " << filename << "\n";
            return 1;
        }

        std::wstring line;

        while (std::getline(file, line)) {
            for (wchar_t& c : line) {
                if (alphabet_size == 33) {
                    if (is_rus(c)) {
                        if (is_lower_rus(c)) {
                            c = caesar_shift(c, (mode == "encode" ? shift :
-shift), russian_lower);
                        } else {
                            c = caesar_shift(c, (mode == "encode" ? shift :
-shift), russian_upper);
                        }
                    }
                } else if (alphabet_size == 26) {
                    if (is_en(c)) {
                        if (is_lower_en(c)) {
                            c = caesar_shift(c, (mode == "encode" ? shift :
-shift), english_lower);
                        } else {
                            c = caesar_shift(c, (mode == "encode" ? shift :
-shift), english_upper);
                        }
                    }
                } else {
                    std::cerr << "Unsupported alphabet size: " <<
alphabet_size << "\n";
                    return 1;
                }
            }
            std::wcout << line << L"\n"; // Output the processed line
        }

        file.close();
        return 0;
    }
```