**БЛОЧНЫЕ АЛГОРИТМЫ СИММЕТРИЧНОГО ШИФРОВАНИЯ**

**Вариант №2**

отчет о лабораторной работе №4

по дисциплине

*МЕТОДЫ И СРЕДСТВА КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ*

Выполнила \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ст. гр. №230711, Павлова В.С.

Проверила \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

доцент каф. ИБ, Басалова Г.В.

**ХОД РАБОТЫ**

**Задание.** Разработайте программу шифрования/расшифрования файлов следующим методом «сдвиг+гаммирование». На первом этапе шифрования каждый байт сообщения подвергается циклическому сдвигу на 5 бит влево. На втором этапе на каждые 4 байта сообщения, прошедшего первый этап шифрования, накладывается 4-байтовая гамма. Гамма вводится с клавиатуры.

**Листинг 1** – Код программы для шифрования и расшифрования

#include <iostream>

#include <string>

#include <fstream>

#include <vector>

using namespace std;

typedef unsigned char uchar;

void EncryptFile(string path, vector<uchar> gamma)

{

ifstream dataFile(path, ios::binary);

// Определение размера файла

dataFile.seekg(0, ios::end);

int fileSize = dataFile.tellg();

dataFile.seekg(0, ios::beg);

vector<uchar> data(fileSize);

dataFile.read(reinterpret\_cast<char\*>(data.data()), fileSize);

string encOutPath = "C:\\Users\\Вика\\Desktop\\encrypt.txt";

ofstream encryptedFile(encOutPath, ios::binary);

**Листинг 1** – Код программы для шифрования и расшифрования (продолжение)

int gammaIndex = 0;

for (int i = 0; i < data.size(); i++)

{

// Сдвиг на 5 бит влево

data[i] = (data[i] << 5) | (data[i] >> 3);

// Гаммирование

data[i] ^= gamma[gammaIndex];

// Переход к следующему байту гаммы

gammaIndex = (gammaIndex + 1) % gamma.size();

}

encryptedFile.write(reinterpret\_cast<const char\*>(data.data()), data.size());

encryptedFile.close();

dataFile.close();

}

void DecryptFile(string path, vector<uchar> gamma)

{

ifstream dataFile(path, ios::binary);

// Определение размера файла

dataFile.seekg(0, ios::end);

int fileSize = dataFile.tellg();

dataFile.seekg(0, ios::beg);

vector<uchar> data(fileSize);

dataFile.read(reinterpret\_cast<char\*>(data.data()), fileSize);

string decrOutPath = "C:\\Users\\Вика\\Desktop\\decrypt.txt";

ofstream outputFile(decrOutPath, ios::binary);

int gammaIndex = 0;

for (int i = 0; i < data.size(); i++)

{

// Гаммирование

data[i] ^= gamma[gammaIndex];

// Циклический сдвиг на 5 бит вправо

data[i] = (data[i] >> 5) | (data[i] << 3);

// Переход к следующему байту гаммы

gammaIndex = (gammaIndex + 1) % gamma.size();

}

outputFile.write(reinterpret\_cast<const char\*>(data.data()), data.size());

dataFile.close();

outputFile.close();

}

int main()

{

string dataPath = "C:\\Users\\Вика\\Desktop\\encrypt.txt";

string gammaValue;

vector<uchar> gamma;

cout << "Write the gamma (2 symbols): ";

cin >> gammaValue;

// Преобразование ключа в массив байтов

for (int i = 0; i < 2; i++)

{

**Листинг 1** – Код программы для шифрования и расшифрования (продолжение)

gamma.push\_back(gammaValue[i]);

}

// Шифрование

// EncryptFile(dataPath, gamma);

// Расшифрование

// DecryptFile(dataPath, gamma);

return 0;

}