Министерство образования Российской Федерации  
Пензенский государственный университет  
Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2  
по курсу «Методы защиты КС»

**Выполнили:**

студенты группы 17ВВ2

Кальнобрицкий К. В.

Аняев А. Р.

**Приняли:**

доц. Дубравин А. В.

к.т.н., доц. Карамышева Н. С.

Пенза, 2020

**Цель работы:** разработка программы шифрования/дешифрирования двоичных файлов с использованием перестановочного шифра.

**Задание:** разработать программу, выполняющую шифрование и расшифровывание произвольного текстового файла с использованием перестановочного шифра используя в качестве ключа последовательность, соответствующую номеру варианта. Выполнить проверку путем двоичного сравнения исходного файла и фала, полученного после расшифровывания.

Размер блока: 4 байта



**Код программы:**

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <string>

using namespace std;

struct block

{

int wholeBlock; //целый блок

int incompleteBlock; //неполный блок

};

int GetTheNumber(char inputData[], const int SIZE);

void TextEncryption(char inputBuffer[], const int SIZE, const int arrKey[], int number);

void TextDecryption(char inputBuffer[], const int sizeBuffer, const int arrKey[], const int sizeKey, int number);

void WriteToFile(const char inputData[], const int SIZE, string directory);

void DeleteFileContents(string directory);

void WriteToFile(block& inputData);

void ReadToFile(block& inputData);

int main()

{

setlocale(LC\_ALL, "rus");

cout << "[-e] [-d] [диск1:][путь1]имя\_файла1 [диск2:][путь2]имя\_файла2\n"

<< "[имя\_файла1] [имя\_файла2] состоит из латинских символов\n"

<< "\n -e Зашифровать имя\_файла1 имя\_файла2"

<< "\n -d Расшифровать имя\_файла1 имя\_файла2\n\n";

cout << "Введите \"exit\" - для завершении работы программы\n\n";

const int N = 32;

const int arrKey[N] = { 6, 29, 17, 3, 26, 13, 7, 0, 8, 15, 12, 19, 21, 5, 28, 16, 23, 24, 1, 2, 25, 30, 31, 10, 9, 14, 27, 18, 22, 4, 11, 20 };

char arrBuffer[N / 8];

string teamName;

do

{

cin >> teamName;

if (teamName.length() == 2)

{

if (teamName == "-e")

{

string fileName1, fileName2;

cin >> fileName1 >> fileName2;

/\* Шифрование \*/

block b{};

ifstream finE;

finE.open(fileName1, ios\_base::binary);

if (!finE.is\_open())

{

EXIT\_FAILURE;

}

DeleteFileContents(fileName2);

do

{

finE.read(arrBuffer, (N / 8));

if ((int)finE.gcount())

{

for (int i = (int)finE.gcount(); i < (N / 8); i++)

{

arrBuffer[i] = NULL;

}

TextEncryption(arrBuffer, (N / 8), arrKey, GetTheNumber(arrBuffer, (N / 8)));

WriteToFile(arrBuffer, (N / 8), fileName2);

((int)finE.gcount() < (N / 8)) ? b.incompleteBlock = (int)finE.gcount() : b.wholeBlock++;

}

} while (!finE.eof());

finE.close();

WriteToFile(b);

cout << "Операция завершилась успехом\n";

}

else if (teamName == "-d")

{

string fileName1, fileName2;

cin >> fileName1 >> fileName2;

/\* Расшифровывание \*/

ifstream finD;

finD.open(fileName1, ios\_base::binary);

if (!finD.is\_open())

{

EXIT\_FAILURE;

}

DeleteFileContents(fileName2);

block b{};

ReadToFile(b);

do

{

finD.read(arrBuffer, (N / 8));

if (b.wholeBlock--)

{

TextDecryption(arrBuffer, (N / 8), arrKey, N, GetTheNumber(arrBuffer, (N / 8)));

WriteToFile(arrBuffer, (N / 8), fileName2);

if (!b.wholeBlock && b.incompleteBlock)

{

finD.read(arrBuffer, (N / 8));

TextDecryption(arrBuffer, (N / 8), arrKey, N, GetTheNumber(arrBuffer, (N / 8)));

char\* arrDecryption = new char[b.incompleteBlock];

for (int i = 0; i < b.incompleteBlock; i++)

{

\*(arrDecryption + i) = arrBuffer[i];

}

WriteToFile(arrDecryption, b.incompleteBlock, fileName2);

break;

}

}

} while (!finD.eof());

finD.close();

cout << "Операция завершилась успехом\n";

}

else

{

cout << "Ошибка ввода команды...\n";

}

}

else if (teamName != "exit")

{

cout << "Ошибка ввода команды...\n";

}

} while (teamName != "exit");

return 0;

}

/\* Получить целое число \*/

int GetTheNumber(char inputData[], const int SIZE)

{

int number = 0;

for (int i = 0; i < (SIZE - 1); i++)

{

number += (int)(unsigned char)inputData[i];

number <<= 8;

}

number += (int)(unsigned char)inputData[SIZE - 1];

return number;

}

/\* Шифрование + запись в буфер \*/

void TextEncryption(char inputBuffer[], const int SIZE, const int arrKey[], int number)

{

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

{

inputBuffer[i] = 0;

for (int j = 0; j < 7; j++)

{

inputBuffer[i] += (int)(bool)((1 << arrKey[8 \* i + j]) & number);

inputBuffer[i] <<= 1;

}

inputBuffer[i] += (int)(bool)((1 << arrKey[8 \* i + 7]) & number);

}

}

/\* Расшифрование + запись в буфер \*/

void TextDecryption(char inputBuffer[], const int sizeBuffer, const int arrKey[], const int sizeKey, int number)

{

for (int index, i = 0; i < sizeBuffer; i++)

{

inputBuffer[i] = 0;

for (int j = 0; j < 7; j++)

{

index = 0;

for (; arrKey[index] != ((sizeKey - 1) - (8 \* i + j)); index++);

inputBuffer[i] += (int)(bool)((1 << (sizeKey - 1 - index)) & number);

inputBuffer[i] <<= 1;

}

index = 0;

for (; arrKey[index] != ((sizeKey - 1) - (8 \* i + 7)); index++);

inputBuffer[i] += (int)(bool)((1 << (sizeKey - 1 - index)) & number);

}

}

/\* Запись в файл \*/

void WriteToFile(const char inputData[], const int SIZE, string directory)

{

ofstream fout;

fout.open(directory, ios\_base::binary | ios\_base::app);

if (!fout.is\_open())

{

EXIT\_FAILURE;

}

fout.write(inputData, SIZE);

fout.close();

}

/\* Очистить файл \*/

void DeleteFileContents(string directory)

{

ofstream fout;

fout.open(directory, ios\_base::binary | ios\_base::trunc);

if (!fout.is\_open())

{

EXIT\_FAILURE;

}

fout.close();

}

/\* Запись блока в файл \*/

void WriteToFile(block& inputData)

{

ofstream fout;

fout.open("block.txt");

if (!fout.is\_open())

{

EXIT\_FAILURE;

}

fout << inputData.wholeBlock << " " << inputData.incompleteBlock;

fout.close();

}

/\* Чтение блока из файл \*/

void ReadToFile(block& inputData)

{

ifstream fin;

fin.open("block.txt");

if (!fin.is\_open())

{

EXIT\_FAILURE;

}

fin >> inputData.wholeBlock >> inputData.incompleteBlock;

fin.close();

}

**Результат работы программы**

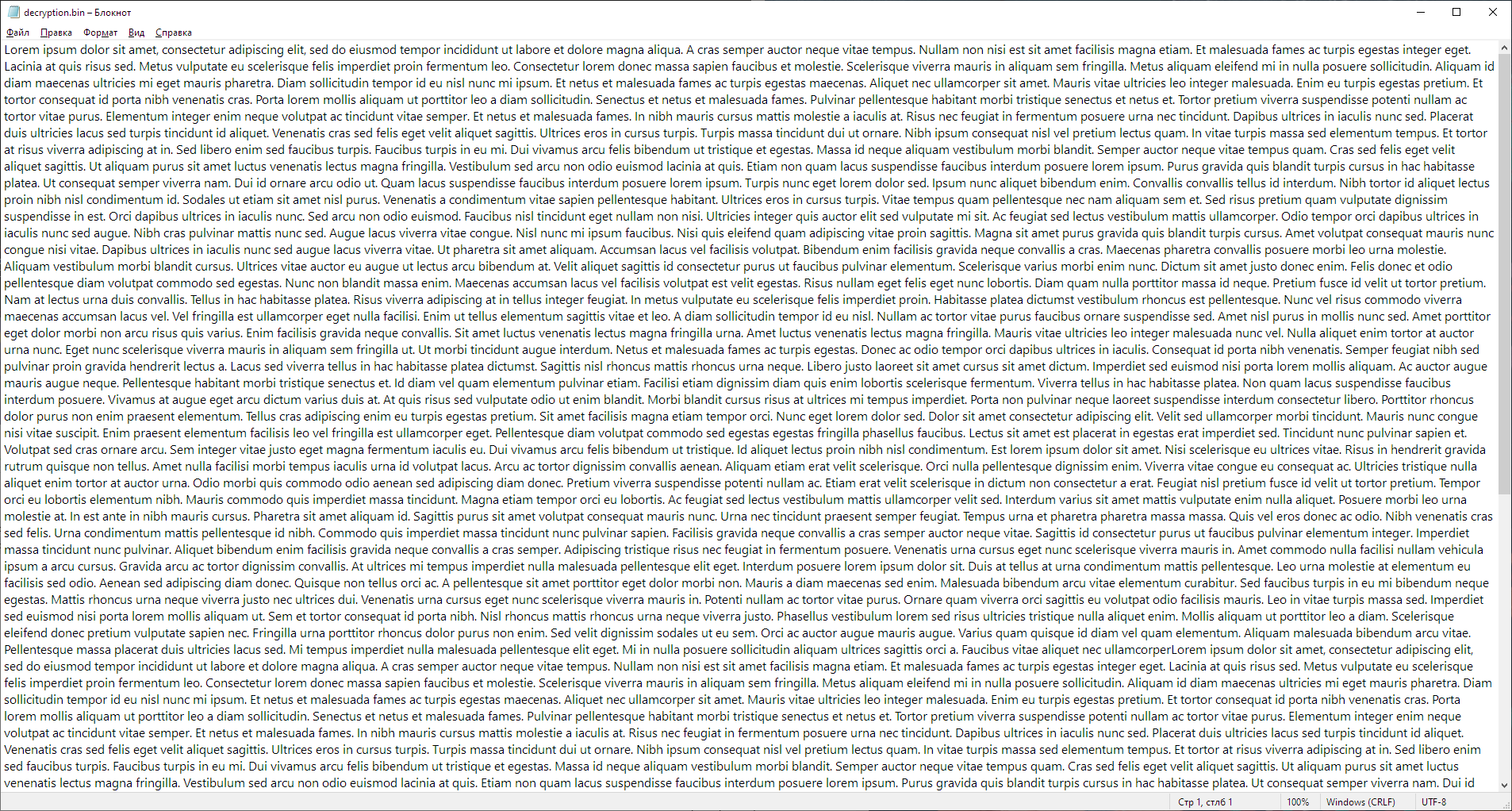
Исходный файл



Зашифрованный файл



Расшифрованный файл



Бинарное сравнение файлов

****

**Вывод:** при выполнении данной лабораторной работы мы научились разрабатывать программы шифрования/дешифрирования текста с использованием перестановочного шифра.