|  |
| --- |
| Пензенский государственный университет  Факультет вычислительной техники  Кафедра «Вычислительная техника» |
| Отчет  по лабораторной работе №1  по дисциплине «Методы защиты компьютерных систем»  Вариант № 1 |
|  |
|  |
| Выполнили: студенты группы 19ВВ3:  Земляков В. Д.  Ерёмин А. А.  Проверил:  Дубравин А. В. |
| Пенза  2022 |

**Тема:** разработка программы шифрования/дешифрирования текста с использованием перестановочного шифра.

Шифрование перестановкой заключается в том, что символы шифруемого текста переставляются по определенному правилу в пределах некоторого блока этого текста. При достаточной длине блока, в пределах которого осуществляется перестановка, и сложном неповторяющемся порядке перестановки можно достигнуть приемлемой стойкости шифра. Ключом является последовательность номеров символов в блоке.

**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 файла, полученного после расшифровывания.

Последовательность перестановки: 3, 6, 5, 2, 4, 1, 7, 9, 8, 10

**Ход работы**

Листинг программы

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int std\_key[10] = {3, 6, 5, 2, 4, 1, 7, 9, 8, 10};

void CopyArray(int\* source, int\* dest, int size)

{

for (size\_t i = 0; i < size; i++)

{

dest[i] = source[i];

}

}

void FilesCompare(string text\_filename, string decode\_filename)

{

ifstream text\_file(text\_filename, ios::binary);

ifstream decode\_file(decode\_filename, ios::binary);

if (text\_file.is\_open() && decode\_file.is\_open())

{

int size\_text;

int size\_decode;

text\_file.seekg(0, text\_file.end);

decode\_file.seekg(0, decode\_file.end);

size\_text = text\_file.tellg();

size\_decode = decode\_file.tellg();

text\_file.seekg(0, text\_file.beg);

decode\_file.seekg(0, decode\_file.beg);

bool result = true;

if (size\_text != size\_decode)

{

result = false;

}

else

{

char ch1, ch2;

do

{

decode\_file.get(ch2);

text\_file.get(ch1);

if (ch1 != ch2)

{

result = false;

break;

}

} while (!text\_file.eof() && !decode\_file.eof());

}

if (result)

cout << "Equal" << endl;

else

cout << "Unequal" << endl;

}

else

cout << "Error opening file!" << endl;

}

class Encoder {

private:

int\* key;

int size\_key;

public:

Encoder()

{

this->size\_key = 10;

this->key = new int[this->size\_key];

CopyArray(std\_key, this->key, this->size\_key);

}

Encoder(int\* key, int size\_key)

{

this->size\_key = size\_key;

this->key = new int[this->size\_key];

CopyArray(key, this->key, size\_key);

}

~Encoder()

{

this->size\_key = 0;

delete[] key;

}

void Encode(char\* source\_text, char\* encode\_text, int size\_source)

{

if (size\_source % this->size\_key != 0)

{

memset(&source\_text[size\_source], ' ', this->size\_key - size\_source);

source\_text[this->size\_key] = '\0';

}

for (size\_t i = 0; i < this->size\_key; i++)

{

encode\_text[i] = source\_text[this->key[i] - 1];

}

}

void Decode(char\* source\_text, char\* decode\_text)

{

for (size\_t i = 0; i < this->size\_key; i++)

{

decode\_text[this->key[i] - 1] = source\_text[i];

}

}

bool FileEncode(string text\_filename, string encode\_filename)

{

ifstream text\_file(text\_filename, ios::binary);

ofstream encode\_file(encode\_filename, ios::binary);

if (text\_file.is\_open() && encode\_file.is\_open())

{

cout << "Files is open. Start Encoding: " << text\_filename << "\n";

int len\_text = 0;

text\_file.seekg(0, text\_file.end);

len\_text = text\_file.tellg();

text\_file.seekg(0, text\_file.beg);

encode\_file << len\_text << ' ';

char\* text = new char[size\_key + 1];

char\* encode\_text = new char[size\_key + 1];

text[size\_key] = '\0';

encode\_text[size\_key] = '\0';

while (!text\_file.eof())

{

text\_file.read(text, this->size\_key);

if (text\_file.eof() && text\_file.gcount() == 0)

break;

Encode(text, encode\_text, text\_file.gcount());

encode\_file.write(encode\_text, this->size\_key);

}

text\_file.close();

encode\_file.close();

cout << "Success! Encode file: " << encode\_filename << "\n";

delete[] text;

delete encode\_text;

}

else

{

cout << "Files opened error!\n";

return 1;

}

}

bool FileDecode(string encode\_filename, string decode\_filename)

{

ifstream encode\_file(encode\_filename, ios::binary);

ofstream decode\_file(decode\_filename, ios::binary);

if (encode\_file.is\_open() && decode\_file.is\_open())

{

cout << "Files is open. Start Decoding " << encode\_filename << "\n";

int len\_text;

encode\_file >> len\_text;

encode\_file.seekg(1, encode\_file.cur);

char\* encode\_text = new char[size\_key + 1];

encode\_text[size\_key] = '\0';

char\* decode\_text = new char[size\_key + 1];

decode\_text[size\_key] = '\0';

while (!encode\_file.eof())

{

encode\_file.read(encode\_text, this->size\_key);

if (encode\_file.eof() && encode\_file.gcount() == 0)

break;

Decode(encode\_text, decode\_text);

int length\_encode = encode\_file.tellg();

if (length\_encode > len\_text)

{

decode\_text[len\_text % size\_key] = '\0';

decode\_file.write(decode\_text, len\_text % size\_key);

break;

}

decode\_file.write(decode\_text, this->size\_key);

}

encode\_file.close();

decode\_file.close();

cout << "Success! Decode file: " << decode\_filename << "\n";

delete[] encode\_text;

delete[] decode\_text;

}

else

{

cout << "Files opened error!\n";

return 1;

}

}

};

int main()

{

Encoder enc = Encoder();

enc.FileEncode("text.txt", "enc\_text.txt");

enc.FileDecode("enc\_text.txt", "dec\_text.txt");

enc.FileEncode("img.png", "enc\_img.png");

enc.FileDecode("enc\_img.png", "dec\_img.png");

}

Результат работы программы

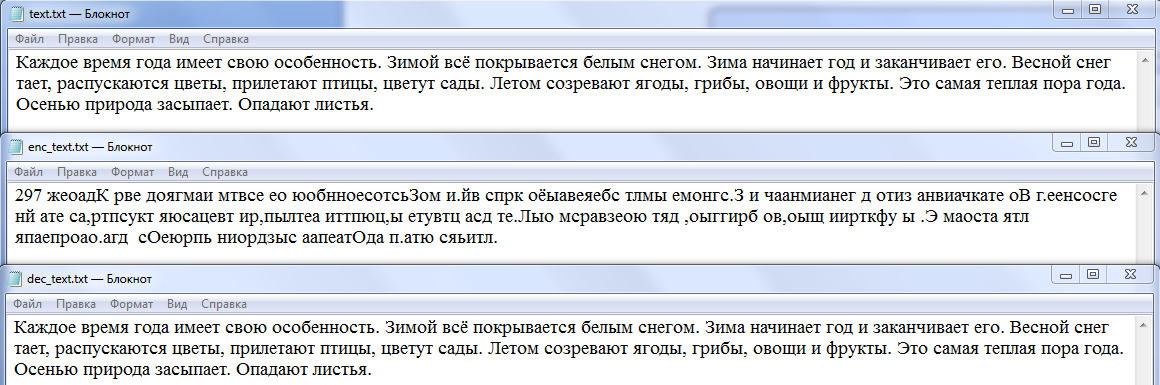


рисунок 1. Сравнение текстовых файлов



рисунок 2. Сравнение текстовых файлов.



рисунок 3. Сравнение изображений.

**Вывод**

В ходе работы познакомились с концепцией перестановочных шифров, написали функции перестановки порядка символов в файле, с помощью которых реализовали программу шифрования/дешифрирования текста с использованием данного шифра. Итоговый и исходный файлы были сопоставлены между собой и оказались одинаковыми.