КРИПТОАНАЛИЗ ШИФРОВ ПЕРЕСТАНОВКИ

Ширинян Л.Х., Горяев В.М. КалмГУ

1.Расшифровка

Процесс расшифрования представляет из себя процесс, обратный шифрованию.

Алгоритм расшифровки:

1. Получаем зашифрованное сообщение.
2. Получаем ключ *k*, которым зашифровано данное сообщение. Переводим в цифровое представление.
3. Делим сообщение на блоки длиной ключа *d*.
4. Переставляем символы в соответствии с ключом.
5. Убираем блоки.
6. Процесс расшифрования окончен.

Данный алгоритм на примере:

1. Пусть имеем следующее предложение для расшифрования:
   ФКЕДА_РИНАРФМАООЦННИ_ЫТЕХОХЛОНЙГ__И
2. **Ключ *k* = «ГАММА»**. Цифровое представление ***k* = 3 1 4 5 2.**
3. Делим предложение на блоки по ***d* = 5** символов:
   | ФКЕДА | _РИНА | РФМАО | ОЦННИ | _ЫТЕХ | ОХЛОН | ЙГ__И |
4. Согласно ключу, первый символ перемещается на третье место, второй символ – на первое место, третий – на четвертое, четвертый – на пятое, пятый – на второе. Получаем:
   | КАФЕД | РА_ИН | ФОРМА | ЦИОНН | ЫХ_ТЕ | ХНОЛО | ГИЙ__ |
5. Убираем блоки и лишние пробелы:
   КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
   Иллюстрация этого примера представлена на рисунке.

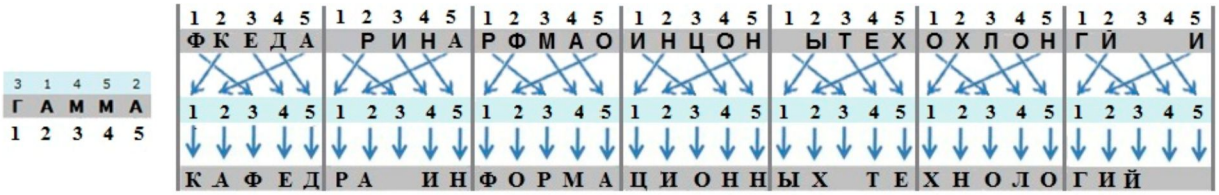


Рис. 1. Схема расшифровки

6. На этом процесс расшифровки завершен.

2. Криптоанализ

Процесс *дешифрования* (криптоанализа) отличается от *расшифровки*. В случае расшифровки подразумевается прочтение зашифрованной информации путём применения заранее определенного алгоритма с использованием имеющегося ключа. А под дешифрованием подразумевается *взлом* шифра, то есть попытка вскрытия и прочтения зашифрованного сообщения третьим лицом, не обладающим ключом.

Если известно, что имеется текст, зашифрованный методом перестановки, но неизвестен ключ перестановки, то дешифровать сообщение можно с помощью перебора всех возможных ключей. Предлагается использовать следующий алгоритм дешифрования:

1. Имеется некоторое зашифрованное сообщение методом перестановки.
2. Пусть длина ключа $d = 2$.
3. Если длина текста не кратна d, добавляем в конец нужное количество пробелов для кратности.
4. Выполняем перебор всевозможных вариантов перестановки по d символов, их будет $d!$ вариантов.
5. Если среди этих вариантов сообщение невозможно прочитать, то $d := d+1$ и переходим к пункту 3. Если получен читаемый текст, то переходим к пункту 6.
6. Получен набор чисел (ключ), которым можно прочесть сообщение.
7. Конец алгоритма.

Процесс дешифровки на примере:

1. Имеется некоторое произвольное закодированное предложение:

   « эотдперелжо неи дялттсевиоряаинг латоир м а»

2. $d = 2$.
3. Длина текста $X = 44$, число 44 кратно 2.
4. Проводим процесс перебора 2! = 2 :

k = 12 эотдперелжо неи дялттсевиоряаинг латоир м а

k = 21 э топдрелеожн иед ляттесивроаяни галотрим а

5. Ни одно из полученных предложений невозможно прочитать => $d := d+1$, т.е. $d = 3$. Возвращаемся к пункту 3:
3. Длина текста $X = 44$. 44 не кратно 3, добавляем 1 пробел.
4. При $d = 3$ количество переборов будет 3! = 6:

k = 123 эотдперелжо неи дялттсевиоряаинг латоир м а

k = 132 оэтпдеерлож енид ятлтесвоираяигн алтиорм а

k = 213 э одтпреежлон е идлятстеивояранигл аоти рма

k = 231 о эптдееролже нди тялетсовиарягина литомр а

k = 312 эо дптреежолне дилтясетиовяарнгила оит мра

k = 321 оэ пдтереожлен д итляестоиваяргниал иотм р а

5. Полученные предложения невозможно прочитать => $d := d+1 = 4$. Возвращаемся к пункту 3:
3. Длина текста $X = 45$. 45 не кратно 4, добавляем 3 пробела.
4. При $d = 4$, количество различных ключей равно 4! = 24 :

k = 1243 этодпреелож ние длятtесвирояаниг алтори ма

k = 1324 оэтдепрежло ени ядлтстевоиряиангл атиор ма

k = 1342 тэодрпееолж ине лдятетсвриоянаига лтрои ам

k = 1423 отэдерпежол еин ялдтсетворияинагла тиро ам

k = 1432 тоэдрепеожл иен лядтествроияниагал трио а м

k = 2134 э отпдерлежон еид ялттсеивораяин глаотирм а

k = 2143 э топдрелеожн иед ляттесивроаяни галотрим а

k = 2314 о этедпржелое ния длсттеовирияанлг аитор ма

k = 2341 т эордпеоелжи нел дяеттсрвионяаиаг лртоиа м

k = 2413 о тэедрпжеоле иня лдстетоврииияналга итро ам

k = 2431 т оэрдепоежли енл ядетстрвоиняиаагл ртиоа м

k = 3124 эо тпедрлжеоне идя лтстеиовраиян лгаоитрм а

k = 3142 эт опрделоежни едл ятетсирвоаняи аглортима

k = 3214 оэ тепдржлеоен ияд лсттеоивриаянл гаиотр м а

k = 3241 тэ орпдеолежин елд яеттсривонаяиа глротиам

k = 3412 от эердпжоелеи нял дсетторвииняалаг ирто а м

k = 3421 то эредпожелие нля десттровинияаалг ритоа м

k = 4123 эот пердлжоенеи дял тсетиорваиня лагоиртм а

k = 4132 это предложение для тестирования алгоритма

k = 4213 оэт епрджлоеени ядл стетоирвианял агиорт ма

k = 4231 тэо рпедолжеине лдя етстриовнаияа лгроитам

k = 4312 отэ ерпджолееин ялд сетторивинаяла гирот ам

k = 4321 тоэ репдожлеиен ляд есттроивниаяал гриота м

5. Одно из полученных последовательностей символов можно прочитать. Само сообщение «это предложение для тестирования алгоритма».
6. Найден ключ перестановки k = **4 1 3 2** .
7. Конец алгоритма.

3. Программная реализация на языке C++ в среде Visual Studio

1. Компилируемый язык со статической типизацией
2. Сочетание высокоуровневых и низкоуровневых средств
3. Реализация объектно-ориентированного программирования
4. STL - стандартная библиотека шаблонов в языке программирования C++.

Входные, выходные данные приложения:

input.txt – входной файл для шифрования;

input_izm.txt – входной файл, дополненный пробелами для кратности;

encode.txt – зашифрованный текст;

decode.txt – расшифрованный текст;

Short_Encode.txt – зашифрованный текст, урезанный до *d*a*, где *d* – длина ключа, *a* – некоторое количество символов. Изначально *a* = 10;

ZeroDecode.txt – файл дешифровки с *1! + 2! + … + n!* результатами;

biblioteka.txt – набор часто используемых английских и русских слов для сравнения с результатами дешифровки.

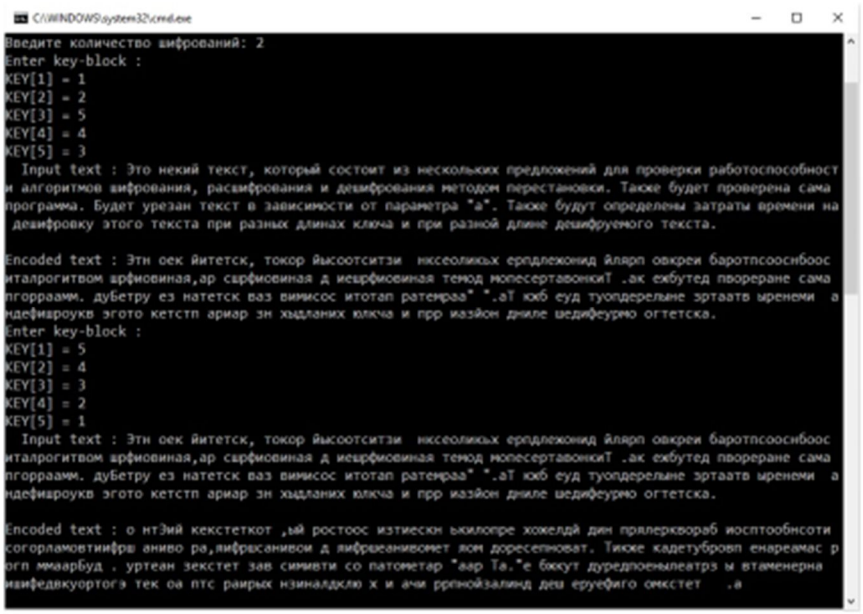


Рис.2. Шифровка сложной перестановки

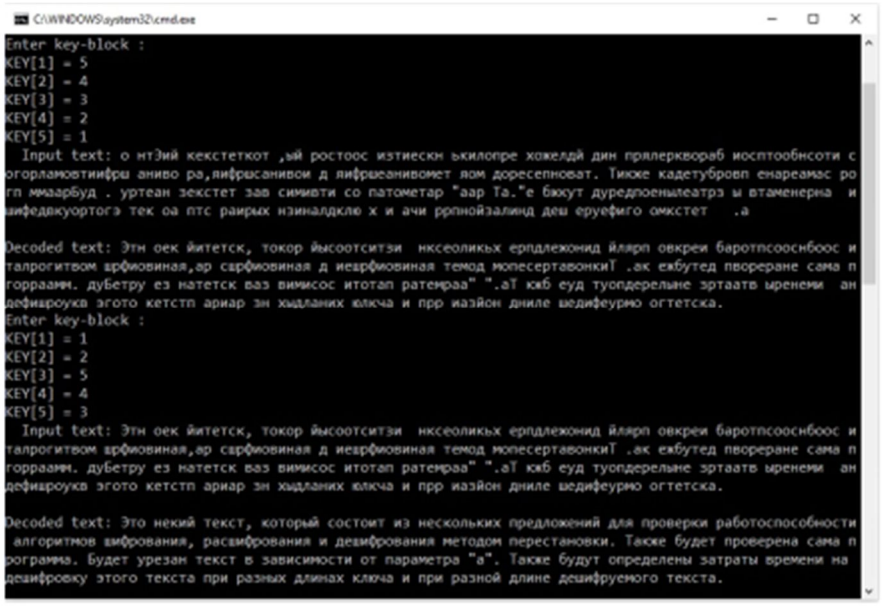


Рис.3. Расшифровка сложной перестановки

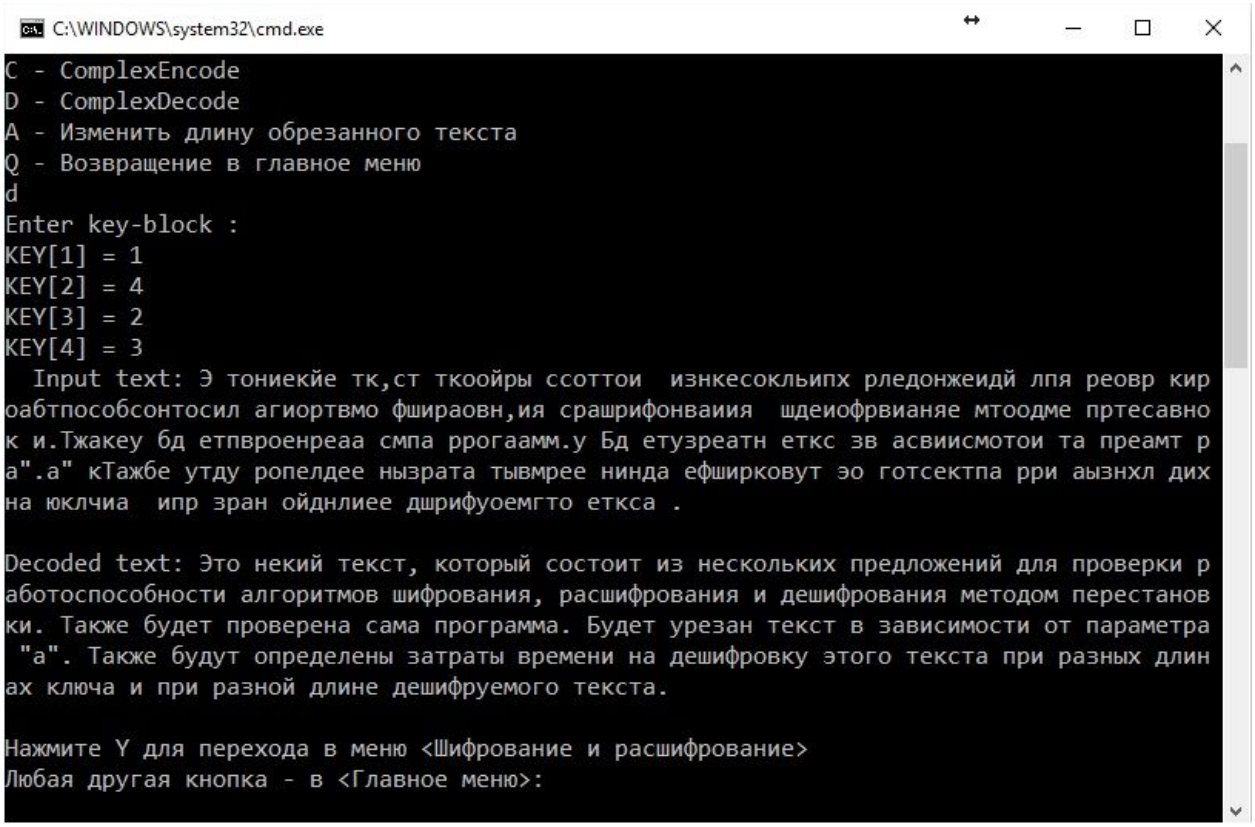
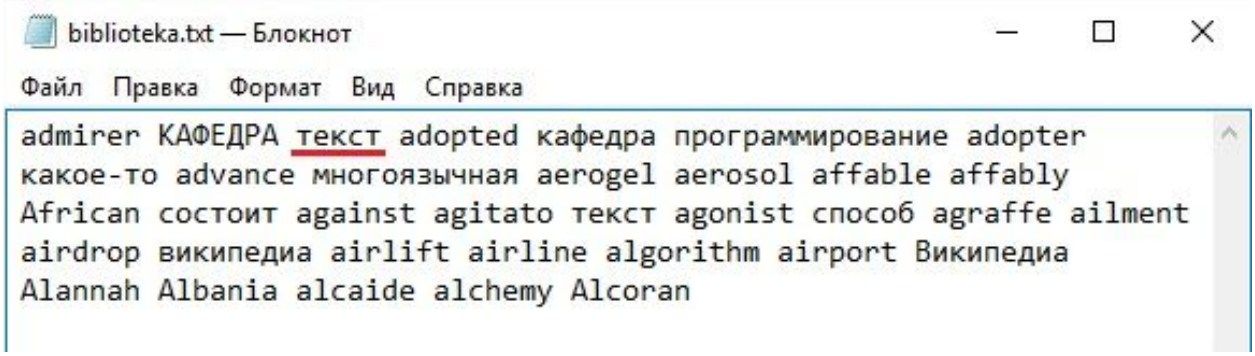
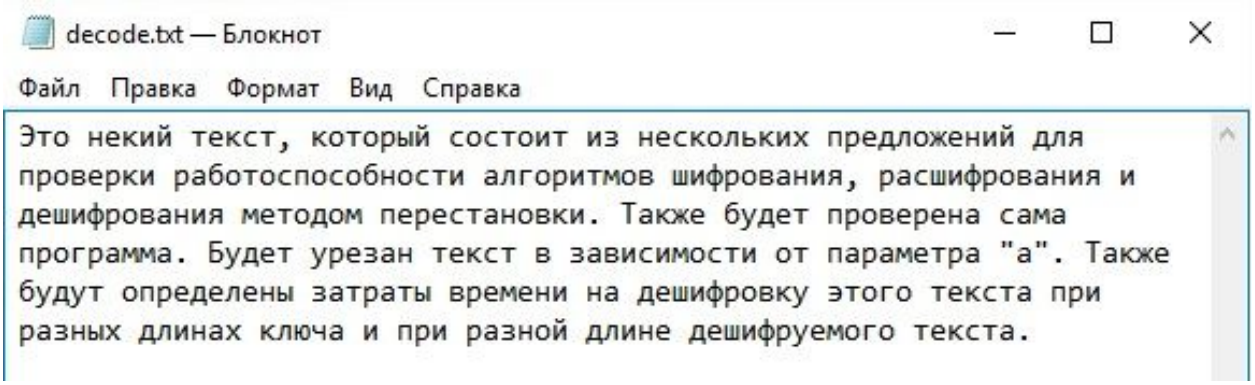


Рис.4. **Расшифровка после дешифрования**

Библиотека *biblioteka.txt*



Расшифрованный текст в decode.txt



Заключение

В данной статье рассмотрены реализованные на C++ алгоритмы криптоанализа шифрами простой и сложной перестановки с промежуточным процессом восстановления ключа шифрования.

Литература:

1. Основы криптографии: учебное пособие / А.П. Алферов [и др.]. – Москва: Издательство Гелиос АРВ, 2002 – 480С.
2. Бабаш, А.В. Криптография: учебное пособие / А.В Бабаш, Шанкин Г.П. – Москва: Издательство Солон-Р, 2002 – 511С.
3. Пилиди В.С. Криптография. Вводные главы: учебное пособие / В.С. Пилиди. –Ростов-на-Дону: Издательство ЮФУ, 2009 – 110С.
4. Харви М. Дейтел, Пол Дж. Дейтел, Как программировать на C++: учебное пособие / Издательство Бином-Пресс, 2010 – 1456С.
5. Бьерн Страуструп, Язык программирования C++: учебное пособие / Издательство Бином-Пресс, 2011 – 1136С.

```
#include <iostream>
#include <fstream>
#include <string>
#include <conio.h>
#include <set>
#include <algorithm>
using namespace std;
//шифрование
string encode(long size, long * Key, string PathF)
{
	string TextF, stroka, encodestr;
	ifstream ifstr(PathF.c_str());
	if (!ifstr)
		cout << "\a\nНе удается открыть файл: " << PathF.c_str() << "\n\a";
	else
	{
	while (getline(ifstr, stroka))
		TextF += stroka;
	ifstr.close();
	cout << "\a  Input text : " << TextF.c_str() << "\n";
	for (long in = 0; in < TextF.length(); in += size)
	{
	stroka = "";
	for (long jm = 0; jm < size; jm++)
		stroka += TextF[in + jm];
	for (long jm = 0; jm < size; jm++)
		encodestr += stroka[Key[jm]];
	}
```

```
        }
        return encodestr;
}
//дешифрование
string decode(long size, long * Key, string PathF)
{
        string TextF, stroka, decodestr;
        ifstream ifsstr(PathF.c_str());
        if (!ifsstr)
                cout << "\a\nНе удается открыть файл: " << PathF.c_str() << "\n\a";
        else
        {
        while (getline(ifsstr, stroka))
                TextF += stroka;
        ifsstr.close();
        cout << "  Input text: " << TextF.c_str() << "\n";
        decodestr = TextF;
        for (long in = 0; in < TextF.length(); in += size)
        {
                for (long jm = 0; jm < size; jm++)
                        decodestr[in + Key[jm]] = TextF[in + jm];
        }
        }
        return decodestr;
}
void clear_file(const string& file_name) //очиста файла
{
        fstream(file_name, fstream::out);
}
//размер файла input.txt
```

```
size_t FileSize(char *FName)
{
	ifstream file(FName);
	size_t size;
	for (size = 0; !file.eof(); size++)
		file.get();
	file.clear();
	file.close();
	return size - 1;
}
fstream ShVix("Short_Encode.txt");
ofstream zero("ZeroDecode.txt");
string text;
int countZero = 1;
void zerodec(int n, int dlina_klyucha)
{
	for (n; n <= dlina_klyucha; n++)
	{
	int kolichestvo = FileSize("Short_Encode.txt"), razn = 0;
	while (kolichestvo % n != 0)
	{
		kolichestvo++;
		razn++;
	}
	for (int cho_to = 0; cho_to < razn; cho_to++)
	{
		ShVix.seekp(0, ios::end) << " ";
	}
	ShVix.close();
	ShVix.open("Short_Encode.txt");
```

```
	long *nKey = new long[n];
	for (int g = 0; g < n; g++)
		nKey[g] = g;
	do{
		cout << "\n   ***KEY*** = ";
		zero << countZero++ << ")   ";
		for (int e = 0; e < n; e++)
		{
			cout << nKey[e] + 1;
				zero << nKey[e] + 1;
		}
		zero << ")   ";
		cout << endl;
		text = decode(n, nKey, "Short_Encode.txt");
		cout << "Decoded text : " << text.c_str() << endl;
		zero << text.c_str() << endl;
	} while (next_permutation(nKey, nKey + n));
	}
}
//копирование из файла в файл
void cp(fstream &f, ofstream &t)
{
	string s((istreambuf_iterator<char>(f)),
		istreambuf_iterator<char>());
	t << s;
}
//поиск строк(и) с выводом
void output_lines(ostream& _out, istream& _in, const set<string>& ws)
{
	const char delim[] = " \t,.!?";
```

```
	string::size_type i, j;
	string s, w;
	bool e = false;
	while (getline(_in, s) && !_in.fail())
	{
	i = 0;
	while ((i = s.find_first_not_of(delim, i)) != string::npos)
	{
	if ((j = s.find_first_of(delim, i)) == string::npos)
		j = s.length();
	if (i != j)
	{
		w.assign(s.begin() + i, s.begin() + j);
		if (ws.find(w) != ws.end())
		{
			cout << "Возможный ключ и фрагмент расшифрованного
сообщения: \n";
			_out << s << endl;;
			e = true;
			break;
		}
	}
		i = j;
	}
	}
	if (!e)
		cout << "Совпадений не найдено\n";
	cout << "D - для продолжения процесса дешифрования\nЛюбая другая
кнопка - переход в меню «Дешифрование»: \n";
}
```

```
//загрузка слов
bool open_words(const char* filename, set<string>& ws){
	ifstream fp(filename);
	if (!fp.is_open())
		return false;
	string s;
	while (fp >> s)
		ws.insert(s);
	fp.close();
	return (ws.size() > 0);
}
void sovpdeniya()
{
	set<string> ws;
	if (!open_words("biblioteka.txt", ws))
		cout << "\nНе удается открыть biblioteka.txt\n ";
	ifstream fp("ZeroDecode.txt");
	cout << endl;
	output_lines(cout, fp, ws);
	ws.clear();
}
int main()
{
…..
}
```