

## Исторические шифры на примере языка С

*Волков Виталий Александрович*

*Мордовский государственный университет им. Н.П. Огарева*

*студент*

### Аннотация

В данной статье рассматриваются некоторые исторические шифры, такие как шифр Цезаря, шифр Виженера, шифр Гронсфельда. Предоставляются примеры применения их на практике. Выполняется их программная реализация. Рассматриваются вопросы надежности таких шифров.

**Ключевые слова:** Шифрование; криптология; криптоанализ; зашифрование; расшифрование; алфавит; шифр; ключ; кодирование; декодирование; дешифрование; криптосистема; шифр Цезаря; шифр Виженера; шифр Гронсфельда.

## Historical ciphers on the example of language С

*Volkov Vitaliy Alexandrovich*

*Ogarev Mordovian State University*

*student*

### Abstract

In this article some historical ciphers, such as Caesar's cipher, Vzhener's cipher, Gronsfeld's cipher are considered. Examples of application them in practice are provided. Their program implementation is executed. Questions of reliability of such ciphers are considered.

**Keywords:** Encoding; cryptology; cryptanalysis; deciphering; alphabet; cipher; key; coding; decoding; cryptosystem; Caesar's cipher; Vzhener's cipher; Gronsfeld's cipher.

Наука, изучающая математические методы защиты информации путем ее преобразования, называется криптология. Криптология разделяется на два направления – криптографию и криптоанализ. Цели этих направлений прямо противоположны.

Криптография занимается поиском и исследованием математических методов преобразования информации с целью сделать ее непонятной для посторонних лиц. Криптоанализ занимается обратной задачей, то есть исследованием возможности расшифрования информации без знания ключей.

В качестве информации, подлежащей шифрованию, будут рассматриваться тексты, построенные на некотором алфавите. По этим терминами понимается следующее:

- Алфавит – конечное множество используемых для кодирования информации знаков.
- Текст (сообщение) – упорядоченный набор из элементов алфавита.
- Зашифрование – процесс преобразования открытых данных в зашифрованные при помощи шифра.
- Расшифрование – процесс преобразования зашифрованных данных в открытые данные при помощи шифра.
- Шифр – совокупность обратимых преобразований множества открытых данных на множество зашифрованных данных, заданных алгоритмом криптографического преобразования.
- Ключ – секретное состояние некоторых параметров алгоритма криптографического преобразования данных, обеспечивающее выбор одного варианта из совокупности всевозможных для данного алгоритма. Секретность ключа должна обеспечивать невозможность восстановления исходного текста по шифрованному.

Вместо термина «открытые данные» часто употребляются термины открытый текст или исходный текст, а вместо «зашифрованные данные» – шифрованный текст или шифротекст.

В некоторых источниках отдельно выделяют термин дешифрование, подразумевая под этим процесс получения исходного текста из шифротекста без знания ключа, то есть методами криптоанализа.

Под шифрованием понимается процесс зашифрования или расшифрования. Также термин шифрование используется как синоним зашифрования. Однако, неверно в качестве синонима шифрования использовать термин «кодирование». Код – правило сопоставления каждому конкретному сообщению строго определённой комбинации символов или сигналов. Процесс преобразования сообщения в комбинацию символов в соответствии с кодом называется кодированием, процесс восстановления сообщения из комбинации символов называется декодированием.

Математически процесс зашифрования можно представить следующей формулой:  $C = E_k(T)$ , где  $T$  – открытый текст,  $E$  – шифрующее преобразование,  $k$  – секретный ключ,  $C$  – шифротекст. Тогда процесс расшифрования запишется в виде:  $T = D_k(C)$ , где  $D$  – расшифровывающее преобразование.

Поскольку смысл любого криптографического преобразования открытого текста состоит в том, чтобы потом этот открытый текст можно было восстановить в первоначальном виде, верно следующее соотношение:  $D_k(E_k(T)) = T$ , при любых допустимых значениях  $T$  и  $k$ .

Заметим, что алгоритмы  $E$  и  $D$  открыты, и секретность исходного текста  $T$  зависит от секретности ключа  $k$ . Обе части этого процесса используют один и тот же ключ, в связи с чем, такие алгоритмы принято называть симметричными криптосистемами, или криптосистемами с секретным ключом. Для пользователей это означает, что прежде чем начать

использовать систему, необходимо получить общий секретный ключ так, чтобы исключить к нему доступ постороннего лица.

Существуют алгоритмы шифрования, зависящие от двух разных ключей. Один из них доступен всем желающим и называется открытым ключом. С его помощью происходит зашифрование информации. Второй ключ держится в секрете и называется секретным ключом. С его помощью происходит расшифрование информации. Такие криптосистемы называются асимметричными, или криптосистемами с открытым ключом.

Коллекцию классических криптосистем открывает один из самых первых известных типов шифра, называемый шифром Цезаря. Процесс шифрования заключается в замене каждой буквы на другую, стоящую от исходной на определенное число позиций в алфавите в зависимости от значения ключа. Система шифрования Цезаря с ключевым словом является одноалфавитной системой подстановки. Особенностью этой системы является использование ключевого слова для смещения и изменения порядка символов в алфавите подстановки.

Выберем некоторое число  $k$ ,  $0 \leq k \leq 25$ , и слово или короткую фразу в качестве ключевого слова. Желательно, чтобы все буквы ключевого слова были различными. Пусть выбраны слово «diplomat» в качестве ключевого слова и число  $k = 5$ .

Ключевое слово записывается под буквами алфавита, начиная с буквы, числовой код которой совпадает с выбранным числом  $k$ :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
					d	i	p	l	o	m	a	t													

Оставшиеся буквы алфавита подстановки записываются после ключевого слова в алфавитном порядке:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
v	w	x	y	z	d	i	p	l	o	m	a	t	b	c	e	f	g	h	j	k	n	q	r	s	u

Требование, чтобы все буквы ключевого слова были различными не обязательно, необходимо только записывать ключевое слово без повторения одинаковых букв.

Количество ключей в системе Цезаря с ключевым словом равно  $n!$ . Для расшифрования необходимо с использованием известного ключа шифрования определить соответствие исходного и заменяющего алфавита и выполнить обратную подстановку.

Напишем на языке C программу шифрования и дешифрования текста, состоящего из нескольких строк, по системе Цезаря с ключевым словом.

Программный код решения поставленной задачи будет следующим:

```
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
using namespace std;
```

```

const int k = 16; // Числовой ключ
string ch = "Защита информации"; // Ключевое слово
string abc = "абвгдеёжзийклмнопрстуфхцчшщъььэюя\
0123456789 .,!?:\;"- _()\
АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЬЬЭЮЯ"; // Алфавит
открытого текста
int i, j, n, Na, Nt, Mt;
string abc2;
// Для цифрового кодирования нового алфавита
int *code = NULL;
// Для цифрового кодирования исходного текста
int **tcode = NULL;
// Для цифрового кодирования зашифрованного текста
int **ctcode = NULL;
// Для зашифрованного текста
char **crypto = NULL;
//Для дешифрованного текста
char **decrypto = NULL;
void NewAbc(void);
int main()
{
    setlocale(LC_ALL, "rus");
    ifstream fin("input.txt");
    if (!fin.is_open())
    {
        cout << "Файл input.txt не может быть открыт" << endl;
        return 0;
    }
    ofstream fout("output.txt");
    if (!fout.is_open())
    {
        cout << "Файл output.txt не может быть открыт" << endl;
        return 0;
    }
    string buf;
    vector<string> text;
    while (getline(fin, buf))
    {
        text.push_back(buf);
    }
    NewAbc();
    cout << "\n ШИФРОВАНИЕ ПО СИСТЕМЕ ЦЕЗАРЯ С КЛЮЧЕВЫМ
СЛОВОМ" << "\n Специальный алфавит с ключевым словом:\n" << abc2 <<
endl;
    Na = abc2.length();
    cout << "\n Мощность специального алфавита:" << Na << " символов\n"
<< endl;
    cout << " Цифровой ключ: k = " << k << "\n Ключевое слово: " << ch <<
endl << endl;

```

```

// Определение количества строк заданного текста
Nt = text.size();
// Определение максимальной длины строки заданного текста
Mt = text[0].length(); // Предполагаемый максимум
for (i = 1; i < Nt; i++)
{
    if (Mt < text[i].length())
        Mt = text[i].length();
}
cout <<
"===== " << endl;
// Выделение памяти для цифрового кодирования нового алфавита
code = (int *)calloc(Na, sizeof(int));
// Цифровое кодирование специального алфавита
for (i = 0; i < Na; i++)
    code[i] = i;
char c;
while (2 > 1)
{
    cout << "Выберите действие:\n1. Зашифрование\n2.
Расшифрование\n3. Выход\n";
    cin >> c;
    switch (c)
    {
    case '1':
        // Для цифрового кодирования исходного текста
        tcode = (int **)calloc(Nt, sizeof(int *));
        for (i = 0; i < Nt; i++)
            tcode[i] = (int *)calloc(Mt, sizeof(int));
        // Для зашифрованного текста
        crypto = (char **)malloc(Nt * sizeof(char *));
        for (i = 0; i < Nt; i++)
            crypto[i] = (char *)malloc((Mt + 1) * sizeof(char));
        // Цифровое кодирование текста
        for (i = 0; i < Nt; i++)
            for (j = 0; j < text[i].length(); j++)
                for (n = 0; n < Na; n++)
                    {
                        if (text[i][j] == abc2[n])
                            tcode[i][j] = code[n];
                    }
        // Шифрование текста
        for (i = 0; i < Nt; i++)
        {
            for (j = 0; j < text[i].length(); j++)
            {
                n = (tcode[i][j] + k) % Na;
                crypto[i][j] = abc2[n];
            }
        }
    }
}

```

```

crypto[i][j] = '\0'; // завершение строки нулевым
СИМВОЛОМ
}
// Вывод зашифрованного текста на консоль
cout << "\n===== Шифротекст:" << endl;
for (i = 0; i < Nt; i++)
{
    cout << crypto[i] << endl;
    fout << crypto[i] << endl;
}
cout <<
"\n===== " << endl;
for (i = 0; i < Nt; i++)
    free(tcode[i]);
free(tcode);
for (i = 0; i < Nt; i++)
    free(crypto[i]);
free(crypto);
break;
case '2':
// Для цифрового кодирования зашифрованного текста
ctcode = (int **)calloc(Nt, sizeof(int *));
for (i = 0; i < Nt; i++)
    ctcode[i] = (int *)calloc(Mt, sizeof(int));
//Для дешифрованного текста
decrypto = (char **)malloc(Nt * sizeof(char *));
for (i = 0; i < Nt; i++)
    decrypto[i] = (char *)malloc((Mt + 1) * sizeof(char));
// Цифровое кодирование зашифрованного текста
for (i = 0; i < Nt; i++)
    for (j = 0; j < text[i].length(); j++)
        for (n = 0; n < Na; n++)
            {
                if (text[i][j] == abc2[n])
                    ctcode[i][j] = code[n];
            }
// Процесс дешифрования
for (i = 0; i < Nt; i++)
{
    for (j = 0; j < text[i].length(); j++)
    {
        n = (ctcode[i][j] + Na - k) % Na;
        decrypto[i][j] = abc2[n];
    }
    decrypto[i][j] = '\0'; // завершение строки нулевым
СИМВОЛОМ
}
}

```

```

        cout << "\n===== Расшифрованный текст:" <<
endl;
        for (i = 0; i < Nt; i++)
        {
            cout << " " << decrypto[i] << endl;
            fout << decrypto[i] << endl;
        }
        cout <<
"\n===== " << endl;
        for (i = 0; i < Nt; i++)
            free(ctcode[i]);
        free(ctcode);
        for (i = 0; i < Nt; i++)
            free(decrypto[i]);
        free(decrypto);
        break;
    case '3':
        return 0;
    default:
        cout << "Выбрано неверное действие!\n\n";
        break;
    }
}
// Освобождение памяти
free(code);
fin.close();
fout.close();
cout << "\n ... Нажмите любую клавишу: " << endl;
getchar();
return 0;
}
void NewAbc(void)
{
    int Nabc = abc.length();
    int Nch = ch.length();
    bool check = false;
    abc2 = string(Nabc, 'x');
    // Запись ключевого слова в алфавит
    n = k;
    for (i = 0; i < Nch; ++i)
    {
        check = false;
        for (j = 0; j < Nabc; ++j)
        {
            if (abc2[j] == ch[i])
            {
                check = true;
                break;
            }
        }
    }
}

```

```

        if (check == false)
        {
            abc2[n++] = ch[i];
        }
    }
    // Запись в алфавит оставшихся букв
    for (i = 0; abc[i] != '\0'; ++i)
    {
        check = false;
        for (j = 0; j < Nch; ++j)
        {
            if (abc[i] == ch[j])
            {
                check = true;
                break;
            }
        }
        if (check == false)
        {
            if ((n) >= Nabc)
            {
                n = 0;
                --i;
            }
            else
                abc2[n++] = abc[i];
        }
    }
}
}
}

```

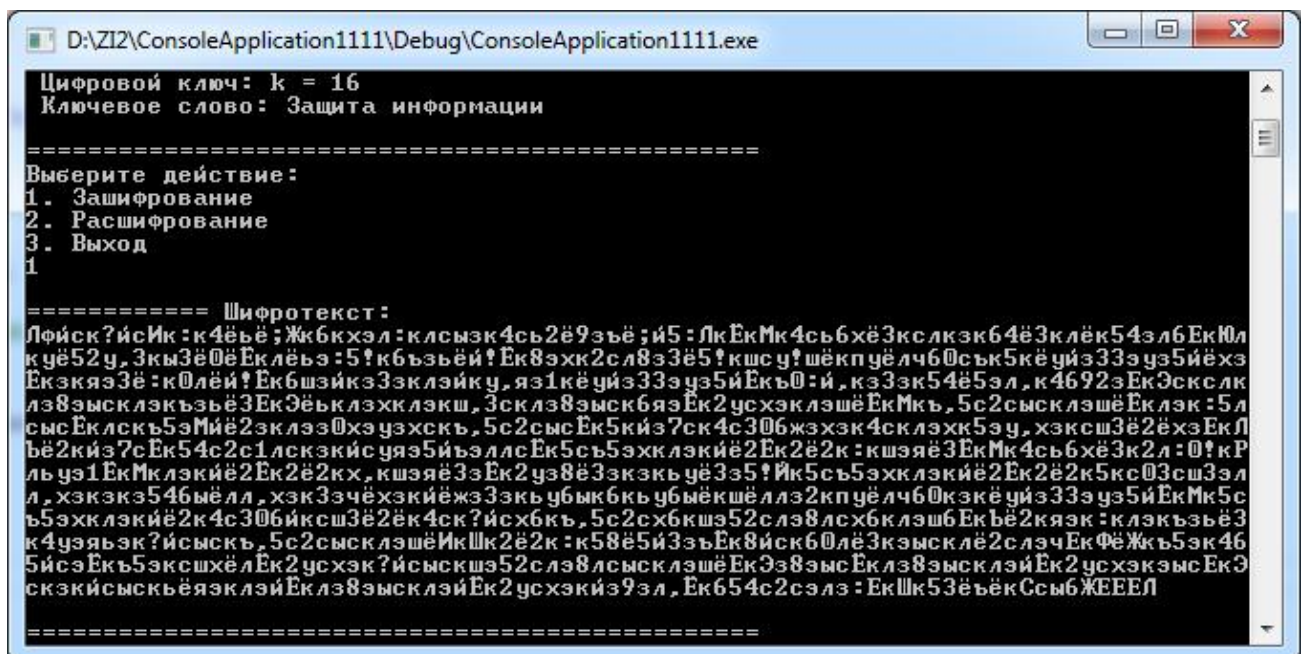


Рисунок 1 – Зашифрование исходного текста шифром Цезаря



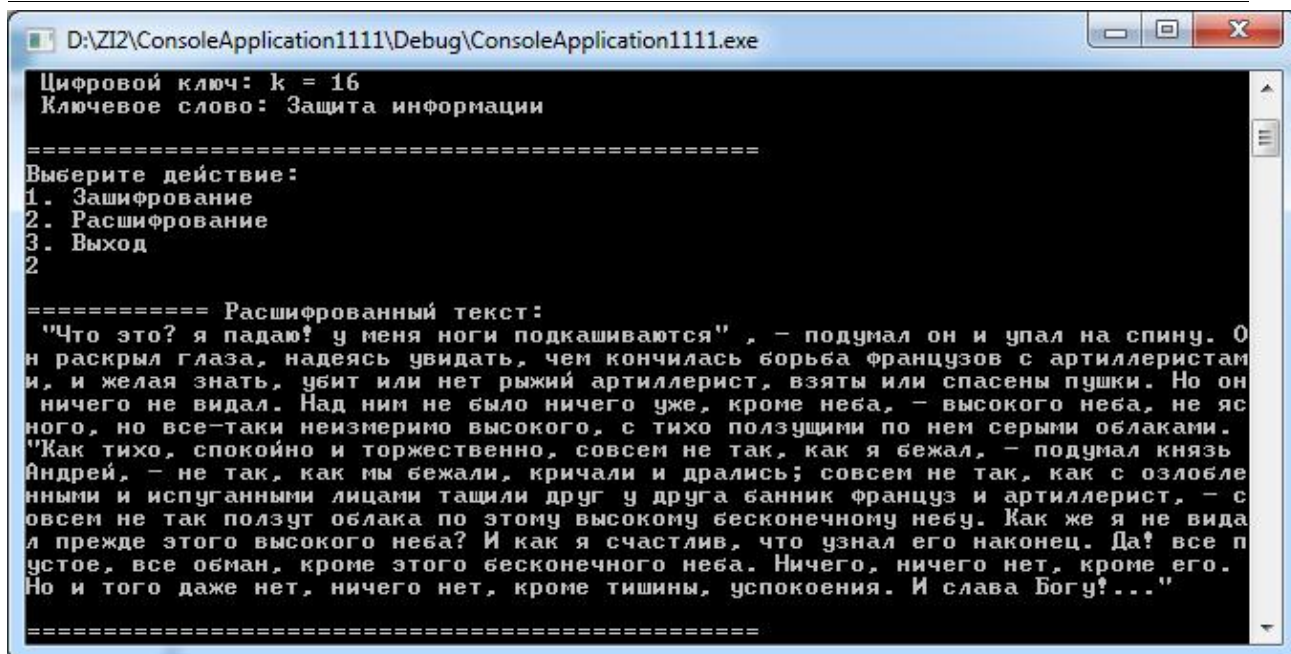


Рисунок 2 – Расшифрование зашифрованного текста

Основной недостаток шифров сдвига и замены заключается в том, что каждая буква открытого текста при шифровании заменяется раз и навсегда фиксированным символом. Поэтому при взломе шифра эффективно работает частотный анализ.

Шифр замены относится к так называемым моноалфавитным шифрам замены, в которых используется только один упорядоченный набор символов, подменяющий собой стандартный алфавит. Один из путей решения указанной проблемы состоит в том, чтобы брать несколько наборов символов вместо стандартного алфавита и шифровать буквы открытого текста, выбирая соответствующие знаки из разных наборов в определенной последовательности. Шифры такого типа носят название полиалфавитных шифров замены.

Например, можно рассмотреть такое соответствие:

```

a b c d e f g h i j k l m n o p q r s t u v w x y z
t m k g o y d s i p e l u a v c r j w x z n h b q f
d c b a h g f e m l k j i z u x w v u t s r q p o n

```

В нем первая строка – латинский алфавит, а вторая и третья – первый и второй алфавиты шифротекста. В этой ситуации буквы открытого текста, стоящие на нечетных позициях, замещаются соответствующими буквами второй строки, а стоящие на четных – третьей. Таким образом, исходное слово «hello» в шифротексте будет выглядеть как «shljv». При этом буква «l», встречающаяся два раза, замещается разными символами. Тем самым существенно усложняется применение статистических методов при атаке на шифр.

В этом примере за один шаг шифруются две буквы. Следовательно, мы имеем дело с блочным шифром, блок которого равен двум буквам латинского алфавита. На практике можно использовать не два, а вплоть до пяти различных алфавитов шифротекста, многократно увеличивая

пространство ключей. Действительно, легко подсчитать, что если берутся символы из пяти замещающих наборов, то число возможных ключей равно  $(26!)^5 \approx 2^{441}$ . Однако пользователю необходимо помнить, что в этом случае ключ – последовательность из  $26 * 5 = 130$  букв. Естественно, чтобы усложнить жизнь злоумышленнику, вскрывающему шифр, необходимо скрыть количество используемых алфавитов, считая его частью ключа. Но для среднего пользователя начала XIX века такая система шифрования казалась слишком громоздкой, поскольку ключ был слишком большим, чтобы запомнить его.

Несмотря на указанный недостаток, самые известные шифры XIX столетия основывались именно на описанном принципе. Шифр Виженера был одним из вариантов полиалфавитного шифра замены, но имел несложный для запоминания ключ. С одной стороны, шифр Виженера является полиалфавитным блочным шифром, но его можно также отнести и к поточным шифрам, естественно обобщающим шифр сдвига.

Как блочный шифр алгоритм Виженера относится к полиалфавитным шифрам, множество замещающих наборов которого ограничивается 26 циклическими сдвигами стандартного латинского алфавита. Если, например, в нем применяются 5 замещающих алфавитов, то пространство ключей сводится к  $26^5 \approx 2^{23}$  возможностям, а в качестве ключа можно запомнить 5 чисел между 0 и 25.

Однако описание шифра Виженера как поточного шифра более естественно. Как и в случае шифра сдвига, можно снова перенумеровать буквы, начиная с 0. Секретный ключ здесь – это короткая последовательность букв, которое повторяется снова и снова, формируя поток ключей. Шифрование заключается в сложении букв открытого текста с буквами потока ключей. Таким образом, формула для зашифрования имеет вид:  $C_i = (T_i + K_{i \bmod m}) \bmod n$ , где  $C_i$  –  $i$ -й символ шифротекста,  $T_i$  –  $i$ -й символ открытого текста,  $K_{i \bmod m}$  – символ ключа с номером  $(i \bmod m)$ ,  $m$  – количество символов в ключе,  $n$  – мощность алфавита, используемого для записи открытого текста. Соответственно формула для расшифрования имеет вид:  $T_i = (C_i - K_{i \bmod m} + n) \bmod n$ .

Например, если ключом является слово «sesame», то шифрование выглядит так:

+	t	h	i	s	i	s	a	t	e	s	t	m	e	s	s	a	g	e
=	s	e	s	a	m	e	s	e	s	a	m	e	s	e	s	a	m	e
	l	l	a	s	u	w	s	x	w	s	f	q	w	w	k	a	s	i

В этом примере буква «а» замещается различными символами в зависимости от того, на каком месте открытого текста она стоит.

При выполнении шифрования вручную можно воспользоваться следующей таблицей (так называемым квадратом Виженера):

```

a b c d e f g h i j k l m n o p q r s t u v w x y z
b c d e f g h i j k l m n o p q r s t u v w x y z a
c d e f g h i j k l m n o p q r s t u v w x y z a b
d e f g h i j k l m n o p q r s t u v w x y z a b c
e f g h i j k l m n o p q r s t u v w x y z a b c d
f g h i j k l m n o p q r s t u v w x y z a b c d e
g h i j k l m n o p q r s t u v w x y z a b c d e f
h i j k l m n o p q r s t u v w x y z a b c d e f g
i j k l m n o p q r s t u v w x y z a b c d e f g h
j k l m n o p q r s t u v w x y z a b c d e f g h i
k l m n o p q r s t u v w x y z a b c d e f g h i j
l m n o p q r s t u v w x y z a b c d e f g h i j k
m n o p q r s t u v w x y z a b c d e f g h i j k l
n o p q r s t u v w x y z a b c d e f g h i j k l m
o p q r s t u v w x y z a b c d e f g h i j k l m n
p q r s t u v w x y z a b c d e f g h i j k l m n o
q r s t u v w x y z a b c d e f g h i j k l m n o p
r s t u v w x y z a b c d e f g h i j k l m n o p q
s t u v w x y z a b c d e f g h i j k l m n o p q r
t u v w x y z a b c d e f g h i j k l m n o p q r s
u v w x y z a b c d e f g h i j k l m n o p q r s t
v w x y z a b c d e f g h i j k l m n o p q r s t u
w x y z a b c d e f g h i j k l m n o p q r s t u v
x y z a b c d e f g h i j k l m n o p q r s t u v w
y z a b c d e f g h i j k l m n o p q r s t u v w x
z a b c d e f g h i j k l m n o p q r s t u v w x y

```

Для зашифрования сообщения из предыдущего примера находим пересечение t-строки и s-столбца. Получаем первую букву шифротекста «l». Аналогичную процедуру повторяем для остальных символов открытого текста.

Для расшифрования сначала находим столбец, соответствующий первой букве ключа, а именно «s». Затем определяем, в какой строке этого столбца находится первая буква шифротекста «l». Получаем первую букву открытого текста «t».

Шифр Вижинера все же не очень сложно взломать, опираясь на статистические закономерности языка на котором составлен текст.

Напишем на языке C программу шифрования фразы «THIS IS A TEST MESSAGE ...» с помощью рекуррентных формул системы шифрования Виженера.

Программный код решения поставленной задачи будет следующим:

```

#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
// Открытый специальный алфавит

```

```

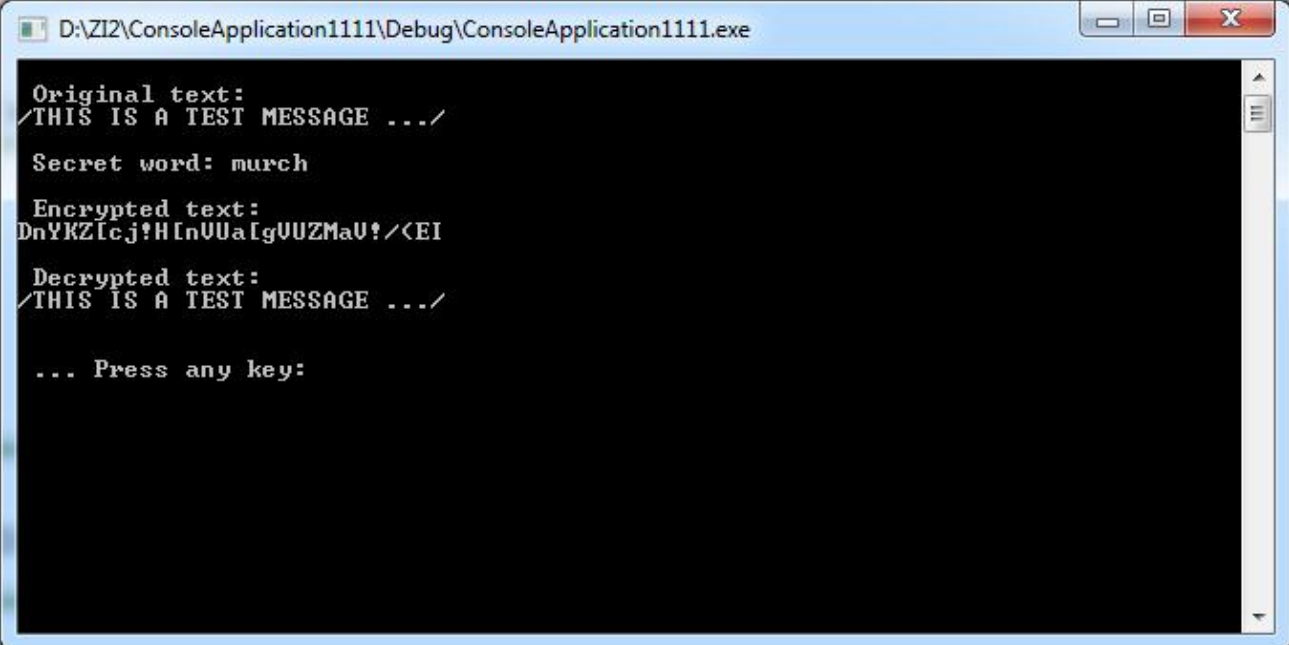
char abc[] = "abcdefghijklmnopqrstuvwxy\
0123456789., :!;/'+_()[]\
ABCDEFGHIJKLMNOPQRSTUVWXYZ";
int main(void)
{
    // Фраза для шифрования
    char str[] = "/THIS IS A TEST MESSAGE .../";
    char secret[] = "murch"; // ключевое слово
    char *crypto, // для зашифрованного текста
        *decrypto; // для дешифрованного текста
    int i, j, n,
        k, // в качестве переменного ключа (сдвига)
        m = strlen(secret); // количество символов лозунга
    int N = (int)strlen(abc);
    // Количество символов заданного текста (фразы)
    int Nt = (int)strlen(str); int *Na, // для цифрового кодирования исходного
    текста
        *Ns, // для цифрового кодирования лозунга
        *Nd; // для цифрового кодирования зашифрованного текста
    // Выделение блоков памяти и запись их адресов в указатели
    crypto = (char *)malloc((Nt + 1) * sizeof(char));
    decrypto = (char *)malloc((Nt + 1) * sizeof(char));
    Na = (int *)malloc(N * sizeof(int));
    Ns = (int *)malloc(m * sizeof(int));
    Nd = (int *)malloc(Nt * sizeof(int));
    // Цифровое кодирование исходного текста
    for (j = 0; j < Nt; j++)
        for (i = 0; i < N; i++)
            if (abc[i] == str[j])
                Na[j] = i;
    // Цифровое кодирование лозунга
    for (j = 0; j < m; j++)
        for (i = 0; i < N; i++)
            if (abc[i] == secret[j])
                Ns[j] = i;
    puts("\n Original text:");
    puts(str);
    printf("\n Secret word: %s\n", secret);
    // Шифрование по методу Виженера
    for (i = 0; i < Nt; i++)
    {
        k = i % m;
        n = (Na[i] + Ns[k]) % N;
        crypto[i] = abc[n];
    }
    crypto[i] = '\0';
    puts("\n Encrypted text:");
}

```

```

puts(encrypt);
// Цифровое кодирование шифрованного текста
for (j = 0; j < Nt; j++)
    for (i = 0; i < N; i++)
        if (abc[i] == crypto[j])
            Nd[j] = i;
// Дешифрование по методу Виженера
for (i = 0; i < Nt; i++)
{
    k = i % m;
    n = (Nd[i] - Ns[k] + N) % N;
    decrypto[i] = abc[n];
}
decrypto[i] = '\0';
puts("\n Decrypted text:");
puts(decrypto);
free(Na);
free(Ns);
free(Nd);
printf("\n\n ... Press any key: ");
_getch();
return 0;
}

```



```

D:\ZI2\ConsoleApplication1111\Debug\ConsoleApplication1111.exe
Original text:
/THIS IS A TEST MESSAGE .../
Secret word: murch
Encrypted text:
DnYKZ[cj?H[nUUa[gUUZMaU?<EI
Decrypted text:
/THIS IS A TEST MESSAGE .../
... Press any key:

```

Рисунок 3 – Работа шифра Виженера

Шифр сложной замены, называемый шифром Гронсфельда, представляет собой модификацию шифра Цезаря с числовым ключом. Для этого под буквами исходного сообщения записывают цифры числового ключа. Если ключ короче сообщения, то его запись циклически повторяют.

Шифротекст получают примерно, как в шифре Цезаря, но отсчитывают по алфавиту не третью букву, а выбирают ту букву, которая смещена по алфавиту на соответствующую цифру ключа. Например, применяя в качестве ключа группу из четырех начальных цифр числа  $e$ , а именно 2718, получаем для исходного сообщения «восточный экспресс» следующий шифротекст:

Сообщение	в	о	с	т	о	ч	н	ы	й	э	к	с	п	р	е	с	с
Ключ	2	7	1	8	2	7	1	8	2	7	1	8	2	7	1	8	2
<u>Шифротекст</u>	д	х	т	ь	р	ю	о	г	л	д	л	щ	с	ч	ж	щ	у

Чтобы зашифровать первую букву сообщения «в», используя первую цифру ключа 2, нужно отсчитать вторую по порядку букву от «в» в алфавите, получается первая буква шифротекста «д».

Следует отметить, что шифр Гронсфельда вскрывается относительно легко, если учесть, что в числовом ключе каждая цифра имеет только десять значений, а значит, имеется лишь десять вариантов прочтения каждой буквы шифротекста. С другой стороны, шифр Гронсфельда допускает дальнейшие модификации, улучшающие его стойкость, в частности двойное шифрование разными числовыми ключами.

Шифр Гронсфельда представляет собой по существу частный случай системы шифрования Виженера.

Напишем на языке C программу шифрования фразы «ВОСТОЧНЫЙ ЭКСПРЕСС №909» с помощью шифра Гронсфельда.

Программный код решения поставленной задачи будет следующим:

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
// Открытый специальный алфавит
string abc = "абвгдеёжзийклмнопрстуфхцчщъьыьэюя. ,:;!?-
+_1234567890№АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧЩЪЬЫЬЭЮЯ";
int main(void)
{
    int i, j, k, m,
        Na, // размер специального алфавита
        Nt, // размер шифруемого текста
        key[] = { 1, 4, 1, 5, 9 }, // ключ - числовой лозунг
        Nk = sizeof(key) / sizeof(key[0]); // размер ключа
    string text;
    string tcode, // указатель на шифрованный текст
        decode; // указатель на расшифрованный текст
    int *Nd, // числовые коды специального алфавита
        *Nkey, // ключи к символам исходного текста
        *dtext, // цифровая кодировка исходного текста
        *dtext2; // цифровая кодировка шифрованного текста
    setlocale(LC_ALL, "rus");
```

```
ifstream fin("input.txt");
if (!fin.is_open())
{
    cout << "Файл input.txt не может быть открыт" << endl;
    getchar();
    return 0;
}
ofstream fout("output.txt");
if (!fout.is_open())
{
    cout << "Файл output.txt не может быть открыт" << endl;
    getchar();
    return 0;
}
// Определение размера специального алфавита
Na = abc.length();
// Числовой код специального алфавита
Nd = (int *)calloc(Na, sizeof(int));
for (i = 0; i < Na; i++)
    Nd[i] = i;
char ch;
while (2 > 1)
{
    cout << "Выберите действие:\n1. Зашифрование\n2.
Расшифрование\n3. Выход\n";
    cin >> ch;
    switch (ch)
    {case '1':          // ***** ЗАШИФРОВАНИЕ *****
        fin.clear();
        fin.seekg(0);
        while (getline(fin, text))
        {
            // Определение размера шифруемого текста
            Nt = text.length();
            Nkey = (int *)calloc(Nt, sizeof(int));
            dtext = (int *)calloc(Nt, sizeof(int));
            dtext2 = (int *)calloc(Nt, sizeof(int));
            // Ключи замены к символам шифруемого текста
            j = 0;
            for (i = 0; i < Nt; i++)
                for (k = 0; k < Nk; k++)
                {
                    if (j < Nt)
                        Nkey[j] = key[k];
                    j++;
                }
            cout << "Строка исходного текста: ";
            cout << text << endl;
            cout << "Ключи замены к строке: ";
            for (i = 0; i < Nt; i++)
```

```
        cout << Nkey[i];
    cout << endl;
    // Цифровое кодирование шифруемого текста
    for (i = 0; i < Nt; i++)
        for (j = 0; j < Na; j++)
            if (text[i] == abc[j])
                dtext[i] = j;
    // Шифрование
    tcode.resize(Nt);
    for (i = 0; i < Nt; i++)
    {
        m = (dtext[i] + Nkey[i]) % (Na);
        tcode[i] = abc[m];
    }
    cout << "Шифрованная строка: ";
    cout << tcode << endl << endl;
    fout << tcode << endl;
    free(dtext);
    free(dtext2);
    free(Nkey);
}
break;
case '2': // ***** РАСШИФРОВАНИЕ *****
    fin.clear();
    fin.seekg(0);
    while (getline(fin, text))
    {
        Nt = text.length(); // Определение размера шифротекста
        Nkey = (int *)calloc(Nt, sizeof(int));
        dtext = (int *)calloc(Nt, sizeof(int));
        dtext2 = (int *)calloc(Nt, sizeof(int));
        // Ключи замены к символам шифруемого текста
        j = 0;
        for (i = 0; i < Nt; i++)
            for (k = 0; k < Nk; k++)
                {
                    if (j < Nt)
                        Nkey[j] = key[k];
                    j++;
                }
        cout << "Шифрованная строка: ";
        cout << text << endl;
        cout << "Ключи замены к строке: ";
        for (i = 0; i < Nt; i++)
            cout << Nkey[i];
        cout << endl;
        // Цифровое кодирование расшифровываемой строки
        for (i = 0; i < Nt; i++)
```



```
        for (j = 0; j < Na; j++)
            if (text[i] == abc[j])
                dtext2[i] = j;
        // Расшифрование
        decode.resize(Nt);
        for (i = 0; i < text.length(); i++)
        {
            m = (dtext2[i] + Na - Nkey[i]) % (Na);
            decode[i] = abc[m];
        }
        cout << "Расшифрованная строка: ";
        cout << decode << endl << endl;
        fout << decode << endl;
        free(dtext);
        free(dtext2);
        free(Nkey);
    }
    break;

case '3':
    return 0;
default:
    cout << "Выбрано неверное действие!\n\n";
    break;
}
}
free(Nd);
cout << "\n\n ... Нажмите любую клавишу: ";
getchar();
return 0;
}
```

