Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Дисциплина «Основы информационной безопасности»

Отчёт по практическому занятию №4.1

**Криптографическая защита информации**

Студент: Жук С.С.

ФИТ 2 курс 2 группа

Преподаватель: Ржеутская Н.В.

**Практическое занятие №4.1**

**«Криптографическая защита информации»**

Цель: Овладение основными криптографическими алгоритмами симметричного шифрования.

**Теоретическое введение**

Криптография - наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним) и аутентичности (целостности и подлинности авторства) информации.

Изначально криптография изучала методы шифрования информации – обратимого преобразования открытого (исходного) текста на основе секретного алгоритма и/или ключа в шифрованный текст (шифротекст). Традиционная криптография образует раздел симметричных криптосистем, в которых зашифрование и расшифрование проводится с использованием одного и того же секретного ключа.

Помимо этого, современная криптография включает в себя асимметричные криптосистемы, системы электронной цифровой подписи, хеш-функции, управление ключами, получение скрытой информации, квантовую криптографию.

Шифрованием (encryption) называют процесс преобразования открытых данных (plaintext) в зашифрованные (шифртекст, ciphertext) или зашифрованных данных в открытые по определенным правилам с применением ключей.

В англоязычной литературе зашифрование / расшифрование – enciphering / deciphering.

Классификация алгоритмов шифрования

1. Симметричные (с секретным, единым ключом, одноключевые, single-key).

1.1. Потоковые:

· с одноразовым или бесконечным ключом (infinite-key cipher);

· с конечным ключом;

· на основе генератора псевдослучайных чисел.

1.2. Блочные:

1.2.1. Шифры перестановки (permutation, P-блоки);

1.2.2. Шифры замены (substitution, S-блоки):

· моноалфавитные;

· полиалфавитные;

2. Асимметричные (с открытым ключом, public-key):

· Диффи-Хеллман DH (Diffie, Hellman);

· Райвест-Шамир-Адлeман RSA (Rivest, Shamir, Adleman);

· Эль-Гамаль (ElGamal).

Симметричные алгоритмы шифрования (или криптография с секретными ключами) основаны на том, что отправитель и получатель информации используют один и тот же ключ. Этот ключ должен храниться в тайне и передаваться способом, исключающим его перехват.

Обмен информацией осуществляется в 3 этапа:

* отправитель передает получателю ключ (в случае сети с несколькими абонентами у каждой пары абонентов должен быть свой ключ, отличный от ключей других пар);
* отправитель, используя ключ, зашифровывает сообщение, которое пересылается получателю;
* получатель получает сообщение и расшифровывает его.

Если для каждого дня и для каждого сеанса связи будет использоваться уникальный ключ, это повысит защищенность системы.

При блочном шифровании информация разбивается на блоки фиксированной длины и шифруется поблочно. Блочные шифры бывают двух основных видов:

· шифры перестановки (transposition, permutation, P-блоки);

· шифры замены (подстановки, substitution, S-блоки).

Шифры перестановок переставляют элементы открытых данных (биты, буквы, символы) в некотором новом порядке. Различают шифры горизонтальной, вертикальной, двойной перестановки, решетки, лабиринты, лозунговые и др.

Шифры замены заменяют элементы открытых данных на другие элементы по определенному правилу. Paзличают шифры простой, сложной, парной замены, буквенно-слоговое шифрование и шифры колонной замены. Шифры замены делятся на две группы:

· моноалфавитные (код Цезаря);

· полиалфавитные (шифр Видженера, цилиндр Джефферсона, диск Уэтстоуна, Enigma).

В моноалфавитных шифрах замены буква исходного текста заменяется на другую, заранее определенную букву. Например, в коде Цезаря буква заменяется на букву, отстоящую от нее в латинском алфавите на некоторое число позиций.



Очевидно, что такой шифр взламывается совсем просто. Нужно подсчитать, как часто встречаются буквы в зашифрованном тексте, и сопоставить результат с известной для каждого языка частотой встречаемости букв.

В полиалфавитных подстановках для замены некоторого символа исходного сообщения в каждом случае его появления последовательно используются различные символы из некоторого набора. Понятно, что этот набор не бесконечен, через какое-то количество символов его нужно использовать снова. В этом слабость чисто полиалфавитных шифров.

В современных криптографических системах, как правило, используют оба способа шифрования (замены и перестановки). Такой шифратор называют составным (product cipher). Oн более стойкий, чем шифратор, использующий только замены или перестановки.

В асимметричных алгоритмах шифрования (или криптографии с открытым ключом) для зашифровывания информации используют один ключ (открытый), а для расшифровывания - другой (секретный). Эти ключи различны и не могут быть получены один из другого.

Схема обмена информацией такова:

· получатель вычисляет открытый и секретный ключи, секретный ключ хранит в тайне, открытый же делает доступным (сообщает отправителю, группе пользователей сети, публикует);

· отправитель, используя открытый ключ получателя, зашифровывает сообщение, которое пересылается получателю;

· получатель получает сообщение и расшифровывает его, используя свой секретный ключ.

**Задание для выполнения.**

1. Изучить теоретические сведения по данной теме.
2. Зашифровать сообщение с использованием шифра
   1. Цезаря,
   2. Трисемуса,
   3. Плейфейра,
   4. Вижинера и полученного секретного ключа (**по номеру варианта** и ключевому слову «**Защита**»). В качестве сообщения использовать свою **Фамилию Имя Отчество**.
3. Расшифровать следующее сообщение:

|  |  |  |
| --- | --- | --- |
| Номер варианта | Сообщение | Способ |
| 10. | Боитдиултоьтдгь-псеоснояшмяил\_ьу-бу\_дччуч\_ | Расшифровать с помощью метода простой перестановки. Таблица 6х7 |

**1) Шифр Цезаря.**

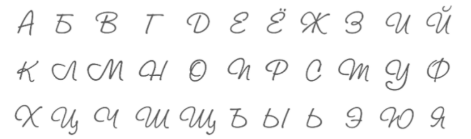
Необходимо зашифровать сообщение «Жук Светлана Сергеевна» с помощью шифра Цезаря и ключа k=3. Исходный алфавит русского языка представлен на рисунке 1.

Рисунок 1 – Алфавит русского языка

Шифр Цезаря один из наиболее древнейших известных шифров. Схема шифрования очень проста - используется сдвиг буквы алфавита на фиксированное число позиций.

Значит, сообщение «Жук Светлана Сергеевна», зашифрованное с использованием шифра Цезаря и заданного ключа k=3, будет иметь следующий вид: «Йцн Фезхогрг Фзуёззерг».

Убедимся в правильности результата, выполнив следующий код:

using System;

public class CaesarCipher

{

//символы русской азбуки

const string alfabet = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";

private string CodeEncode(string text, int k)

{

//добавляем в алфавит маленькие буквы

var fullAlfabet = alfabet + alfabet.ToLower();

var letterQty = fullAlfabet.Length;

var retVal = "";

for (int i = 0; i < text.Length; i++)

{

var c = text[i];

var index = fullAlfabet.IndexOf(c);

if (index < 0)

{

//если символ не найден, то добавляем его в неизменном виде

retVal += c.ToString();

}

else

{

var codeIndex = (letterQty + index + k) % letterQty;

retVal += fullAlfabet[codeIndex];

}

}

return retVal;

}

//шифрование текста

public string Encrypt(string plainMessage, int key)

=> CodeEncode(plainMessage, key);

//дешифрование текста

public string Decrypt(string encryptedMessage, int key)

=> CodeEncode(encryptedMessage, -key);

}

class Program

{

static void Main(string[] args)

{

var cipher = new CaesarCipher();

Console.Write("Введите текст: ");

var message = Console.ReadLine();

Console.Write("Введите ключ: ");

var secretKey = Convert.ToInt32(Console.ReadLine());

var encryptedText = cipher.Encrypt(message, secretKey);

Console.WriteLine("Зашифрованное сообщение: {0}", encryptedText);

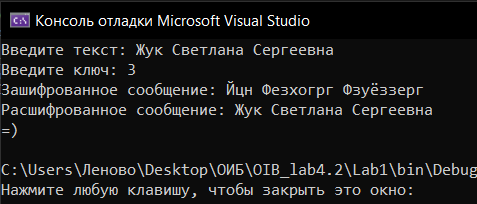
Console.WriteLine("Расшифрованное сообщение: {0}", cipher.Decrypt(encryptedText, secretKey));

Console.ReadLine();

}

}

Выполнение программы:



**2)Шифр Трисемуса**

Необходимо зашифровать сообщение «Жук Светлана Сергеевна» с помощью шифра Трисемуса и ключевого слова «Защита». Таблица Трисемуса представлена на рисунке 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| з | а | щ | и | т | б |
| в | г | д | е | ё | ж |
| й | к | л | м | н | о |
| п | р | с | у | ф | х |
| ц | ч | ш | ъ | ы | ь |
| э | ю | я |  | , | . |

Рисунок 2 – Таблица Трисемуса

Для получения такого шифра замены обычно использовались таблица для записи букв алфавита и ключевое слово (или фраза). В таблицу сначала вписывалось по строкам ключевое слово, причем повторяющиеся буквы отбрасывались. Затем эта таблица дополнялась не вошедшими в нее буквами алфавита по порядку.

Каждая буква открытого сообщения заменяется буквой, расположенной под ней в том же столбце. Если буква находится в последней строке таблицы, то для ее шифрования берут самую верхнюю букву столбца.

Значит, сообщение «Жук Светлана Сергеевна», зашифрованное с использованием шифра Трисемуса и заданного ключевого слова «Защита», будет иметь следующий вид: «ОЪРИШЙМЁСГФГИШМЧКММЙФГ».

Убедимся в правильности результата, выполнив следующий код:

using System;

using System.Linq;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

int n, m;

Console.WriteLine("Введите размерность таблицы");

n = Convert.ToInt32(Console.ReadLine());

m = Convert.ToInt32(Console.ReadLine());

char[] alphabet = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ ,.".ToCharArray();

Console.WriteLine("Символов в алфавите: " + alphabet.Length);

int rows = 0, columns;

bool isValidTable;

do

{

Console.Write("Количество колонок в таблице: ");

isValidTable = int.TryParse(Console.ReadLine(), out columns) && columns > 1;

if (!isValidTable)

{

Console.WriteLine("Необходимо ввести число больше 1");

}

else

{

rows = alphabet.Length / columns;

isValidTable &= rows > 1 && rows \* columns == alphabet.Length;

if (!isValidTable)

{

Console.WriteLine("Необходимо ввести число колонок таким образом, чтобы число строк таблицы было больше 1 и таблица могла вмещать в себе все символы алфавита");

}

}

}

while (!isValidTable);

char[] keyWord;

bool isValidKeyWord;

do

{

Console.Write("Введите ключевое слово: ");

keyWord = Console.ReadLine().ToUpper().Distinct().ToArray();

isValidKeyWord = keyWord.Length > 0 && keyWord.Length <= alphabet.Length;

if (!isValidKeyWord)

{

Console.WriteLine("Ключевое слово не может быть пустой строкой или содержать число уникальных символов больше размера алфавита");

}

else

{

isValidKeyWord &= !keyWord.Except(alphabet).Any();

if (!isValidKeyWord)

{

Console.WriteLine("Ключевое слово не может содержать символы, которых нет в алфавите");

}

}

}

while (!isValidKeyWord);

var table = new char[rows, columns];

for (var i = 0; i < keyWord.Length; i++)

{

table[i / columns, i % columns] = keyWord[i];

}

alphabet = alphabet.Except(keyWord).ToArray();

for (var i = 0; i < alphabet.Length; i++)

{

int position = i + keyWord.Length;

table[position / columns, position % columns] = alphabet[i];

}

string message;

bool isValidMessage;

do

{

Console.Write("Введите сообщение: ");

message = Console.ReadLine().ToUpper();

isValidMessage = !string.IsNullOrEmpty(message);

if (!isValidMessage)

{

Console.WriteLine("Сообщение не может быть пустой строкой");

}

}

while (!isValidMessage);

var result = new char[message.Length];

for (var k = 0; k < message.Length; k++)

{

char symbol = message[k];

for (var i = 0; i < rows; i++)

{

for (var j = 0; j < columns; j++)

{

if (symbol == table[i, j])

{

symbol = table[(i + 1) % rows, j];

i = rows;

break;

}

}

}

result[k] = symbol;

}

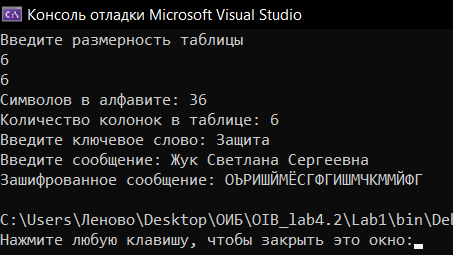
Console.WriteLine("Зашифрованное сообщение: " + new string(result));

}

}

}

Выполнение программы:



**3)Шифр Плейфейра**

Необходимо зашифровать сообщение «Жук Светлана Сергеевна» с помощью шифра Плейфейра и ключевого слова «Защита». Промежуточная биграмма имеет вид «ЖУ-КС-ВЕ-ТЛ-АН-АС-ЕР-ГЕ-ЕВ-НА». Таблица Плейфейра представлена на рисунке 3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| З | А | Щ | И | Т | Б | В | Г |
| Д | Е | Ж | Й | К | Л | М | Н |
| О | П | Р | С | У | Ф | Х | Ц |
| Ч | Ш | Ъ | Ы | Ь | Э | Ю | Я |

Рисунок 3 – Таблица Плейфейра

Шифр предусматривает шифрование пар символов (биграмм). Таким образом, этот шифр более устойчив к взлому по сравнению с шифром простой замены, так как затрудняется частотный анализ. Он может быть проведен, но не для 26 возможных символов (латинский алфавит), а для 26 х 26 = 676 возможных биграмм. Анализ частоты биграмм возможен, но является значительно более трудным и требует намного большего объема зашифрованного текста.

Для шифрования сообщения необходимо разбить его на биграммы (группы из двух символов), при этом, если в биграмме встретятся два одинаковых символа, то между ними добавляется заранее оговоренный вспомогательный символ (Ъ).

Значит, сообщение «Жук Светлана Сергеевна», зашифрованное с использованием шифра Плейфейра и заданного ключевого слова «Защита», будет иметь следующий вид: «КР-ЙУ-АМ-БК-ГЕ-ИП-ЖП-АН-МА-ЕГ», то есть Крй Уамбкгеи Пжпанмаег.

Убедимся в правильности результата, выполнив следующий код:

#include <iostream>

#include <string>

using namespace std;

// Функция для нахождения координаты символа в матрице

void findCharInMatrix(char matrix[4][8], char ch, int& row, int& col) {

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 8; j++) {

if (matrix[i][j] == ch) {

row = i;

col = j;

return;

}

}

}

}

// Функция для зашифрования сообщения

string playfairEncrypt(char matrix[4][8], string message) {

string encrypted = "";

int len = message.length();

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

char ch1 = message[i];

char ch2 = (i + 1 < len) ? message[i + 1] : 'X';

int row1, col1, row2, col2;

findCharInMatrix(matrix, ch1, row1, col1);

findCharInMatrix(matrix, ch2, row2, col2);

if (row1 == row2) {

col1 = (col1 + 1) % 8;

col2 = (col2 + 1) % 8;

}

else if (col1 == col2) {

row1 = (row1 + 1) % 4;

row2 = (row2 + 1) % 4;

}

else {

int temp = col1;

col1 = col2;

col2 = temp;

}

encrypted += matrix[row1][col1];

encrypted += matrix[row2][col2];

}

return encrypted;

}

// Функция для расшифрования сообщения

string playfairDecrypt(char matrix[4][8], string encrypted) {

string decrypted = "";

int len = encrypted.length();

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

char ch1 = encrypted[i];

char ch2 = encrypted[i + 1];

int row1, col1, row2, col2;

findCharInMatrix(matrix, ch1, row1, col1);

findCharInMatrix(matrix, ch2, row2, col2);

if (row1 == row2) {

decrypted += matrix[row1][(col1 + 7) % 8];

decrypted += matrix[row2][(col2 + 7) % 8];

}

else if (col1 == col2) {

decrypted += matrix[(row1 + 3) % 4][col1];

decrypted += matrix[(row2 + 3) % 4][col2];

}

else {

decrypted += matrix[row1][col2];

decrypted += matrix[row2][col1];

}

}

return decrypted;

}

int main() {

setlocale(LC\_ALL, "Russian"); // Русская локализация консоли

char matrix[4][8] = {

{'З', 'А', 'Щ', 'И', 'Т', 'Б', 'В', 'Г'},

{'Д', 'Е', 'Ж', 'Й', 'К', 'Л', 'М', 'Н'},

{'О', 'П', 'Р', 'С', 'У', 'Ф', 'Х', 'Ц'},

{'Ч', 'Ш', 'Ъ', 'Ы', 'Ь', 'Э', 'Ю', 'Я'}

};

string message = "ЖУКСВЕТЛАНАСЕРГЕЕВНА";

string encrypted = playfairEncrypt(matrix, message);

string decrypted = playfairDecrypt(matrix, encrypted);

cout << "Исходное сообщение: " << message << endl;

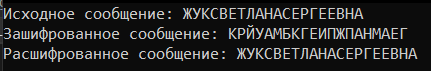
cout << "Зашифрованное сообщение: " << encrypted << endl;

cout << "Расшифрованное сообщение: " << decrypted << endl;

return 0;

}

Выполнение программы:



**4)Шифр Вижинера**

Необходимо зашифровать сообщение «Жук Светлана Сергеевна» с помощью шифра Вижинера и ключевого слова «Защита». Промежуточный ключ имеет вид «защитазащитазащитаза» (человек, посылающий сообщение, записывает ключевое слово («защита») циклически до тех пор, пока его длина не будет соответствовать длине исходного текста). Таблица Вижинера представлена на рисунке 4.

Шифр Виженера состоит из последовательности нескольких шифров Цезаря с различными значениями сдвига. Для зашифровывания может использоваться таблица алфавитов, называемая tabula recta или квадрат (таблица) Виженера. Применительно к латинскому алфавиту таблица Виженера составляется из строк по 32 символов, причём каждая следующая строка сдвигается на несколько позиций. Таким образом, в таблице получается 32 различных шифров Цезаря. На каждом этапе шифрования используются различные алфавиты, выбираемые в зависимости от символа ключевого слова.

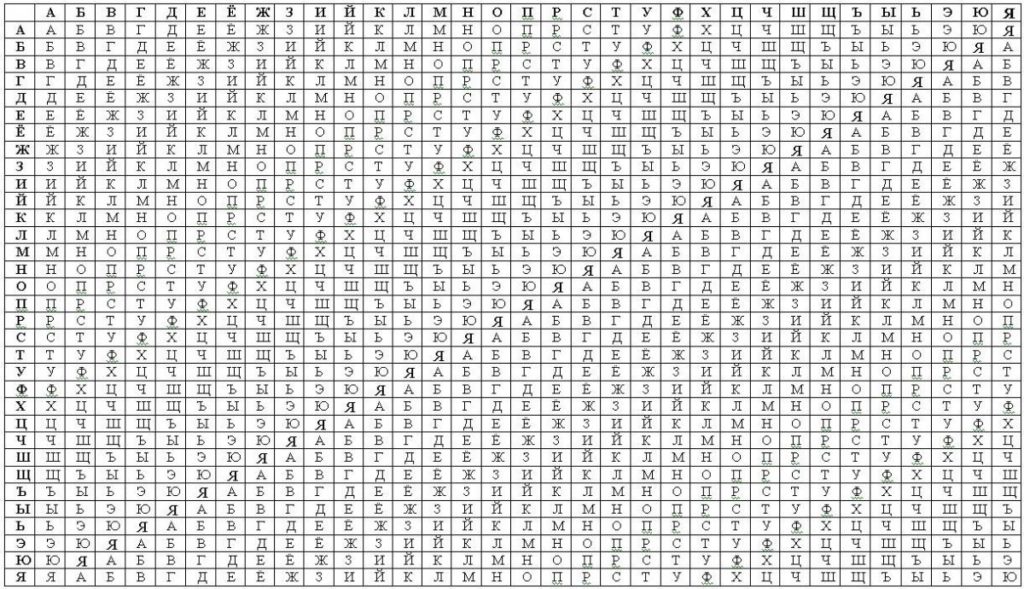


Рисунок 4 – Таблица Вижинера

1) Запишем фразу, которую нужно зашифровать без пробелов: жуксветланасергеевна

2) Циклически запишем кодовое слово таким образом, чтобы его длина совпадала с фразой, которую необходимо зашифровать:

жуксветланасергеевна

защитазащитазащитаза

3) Совместим соответствующие буквы открытого и закрытого ключа для получения шифра: нугщфещлщхтсмрьнчвфа.

Значит, сообщение «Жук Светлана Сергеевна» с промежуточным ключом «защитазащитазащитаза», зашифрованное с использованием шифра Виженера и заданного ключевого слова «Защита», будет иметь следующий вид: «ОУД ЪФЕЪЛЩЦТ СМРЬНЧВХА».

Убедимся в правильности результата, выполнив следующий код:

using System;

public class VigenereCipher

{

const string defaultAlphabet = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";

readonly string letters;

public VigenereCipher(string alphabet = null)

{

letters = string.IsNullOrEmpty(alphabet) ? defaultAlphabet : alphabet;

}

// Генерация повторяющегося пароля

private string GetRepeatKey(string s, int n)

{

var p = s;

while (p.Length < n)

{

p += p;

}

return p.Substring(0, n);

}

private string Vigenere(string text, string password, bool encrypting = true)

{

var gamma = GetRepeatKey(password, text.Length);

var retValue = "";

var q = letters.Length;

for (int i = 0, j = 0; i < text.Length; i++)

{

if (text[i] == ' ')

{

// Пропускаем пробелы и добавляем их в результирующую строку без изменений

retValue += ' ';

continue;

}

var letterIndex = letters.IndexOf(text[i]);

var codeIndex = letters.IndexOf(gamma[j]);

if (letterIndex < 0)

{

// Если буква не найдена, добавляем её в исходном виде

retValue += text[i].ToString();

}

else

{

retValue += letters[(q + letterIndex + ((encrypting ? 1 : -1) \* codeIndex)) % q].ToString();

}

j = (j + 1) % password.Length; // Следующая буква ключа

}

return retValue;

}

// Шифрование текста

public string Encrypt(string plainMessage, string password)

=> Vigenere(plainMessage, password);

// Дешифрование текста

public string Decrypt(string encryptedMessage, string password)

=> Vigenere(encryptedMessage, password, false);

}

class Program

{

static void Main(string[] args)

{

// Передаем в конструктор класса буквы русского алфавита

var cipher = new VigenereCipher();

Console.Write("Введите текст: ");

var inputText = Console.ReadLine().ToUpper();

Console.Write("Введите ключ: ");

var password = Console.ReadLine().ToUpper();

var encryptedText = cipher.Encrypt(inputText, password);

Console.WriteLine("Зашифрованное сообщение: {0}", encryptedText);

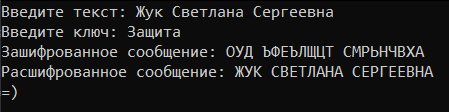
Console.WriteLine("Расшифрованное сообщение: {0}", cipher.Decrypt(encryptedText, password));

Console.ReadLine();

}

}

Выполнение программы:



# **Индивидуальное задание**

|  |  |  |
| --- | --- | --- |
| Номер варианта | Сообщение | Способ |
| 10. | Боитдиултоьтдгь-псеоснояшмяил\_ьу-бу\_дччуч\_ | Расшифровать с помощью метода простой перестановки. Таблица 6х7 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Б | у | д | е | ш | ь | д |
| о | л | г | о | м | у | ч |
| и | т | ь | с | я | \_ | ч |
| т | о | - | н | и | б | у |
| д | ь | п | о | л | у | ч |
| и | т | с | я | \_ | \_ | \_ |

Расшифровка: Будешьдолгомучиться -что-нибудьполучится \_ \_ \_

Убедимся в правильности результата, выполнив следующий код:

#include <iostream>

#include <string>

int main() {

setlocale(LC\_ALL, "Russian"); // Русская локализация консоли

// Исходное сообщение

std::string message = "Боитдиултоьтдгь-псеоснояшмяил\_ьу-бу\_дччуч\_";

// Создаем матрицу 6x7 и заполняем её символами вручную

char matrix[6][7] = {

{'Б', 'у', 'д', 'е', 'ш', 'ь', 'д'},

{'о', 'л', 'г', 'о', 'м', 'у', 'ч'},

{'и', 'т', 'ь', 'с', 'я', '-', 'ч'},

{'т', 'о', '-', 'н', 'и', 'б', 'у'},

{'д', 'ь', 'п', 'о', 'л', 'у', 'ч'},

{'и', 'т', 'с', 'я', '\_', '\_', '\_'}

};

// Выводим расшифрованное сообщение

std::cout << "Расшифрованное сообщение: ";

for (int i = 0; i < 6; i++) {

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

if (matrix[i][j] != '\_') {

std::cout << matrix[i][j];

}

else {

std::cout << ' ';

}

}

}

std::cout << std::endl;

return 0;

}

Выполнение программы:



**Вывод:** в ходе лабораторной работы были изучены основные криптографические алгоритмы симметричного шифрования.