Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Практическое задание 4\_1

По дисциплине «Основы информационной безопасности»

На тему «Криптографическая защита информации»

Выполнила:

Студентка 2 курса 2 группы

Глухова Д.В.

Вариант 6

Преподаватель: ст. пр. Ржеутская Н. В.

2023, Минск

**Практическое занятие №4.1**

**Тема «Криптографическая защита информации»**

Цель: Овладение основными криптографическими алгоритмами симметричного шифрования .

Сообщение (ФИО): Глухова Дарья Витальена

Ключ (Номер варианта): 6

Порядок букв по алфавиту:



**Шифр цезаря**

Исходные буквы:

4 13 21 23 16 3 1 – Глухова

5 1 18 28 33 – Дарья

3 10 20 1 13 28 6 3 15 1 – Витальевна

Сдвиг:

10 19 27 29 22 9 7 – Исщыфзё

11 7 24 3 6 – Йёцве

9 16 26 7 19 3 12 9 21 7 – Зошёсвкзуё

Зашифрованное сообщение: Исщыфзё Йёцве Зошёсвкзуё

**Шифр Трисемуса**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | ь | ы | ъ | э | ю | я |

Нфъючме Оеьщг Мйкефщпмце

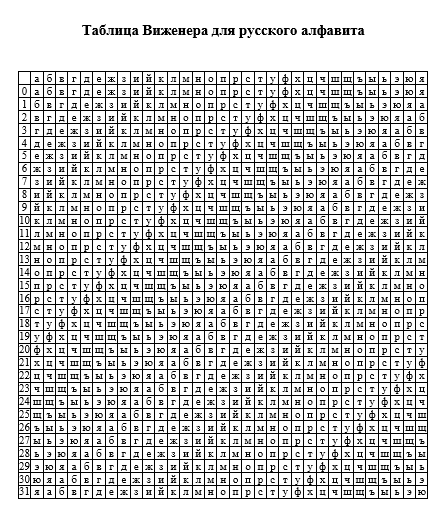
**Шифр Плейфера**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

Гл ух ов аД ар ья Ви та ль ев на

Бн фц хз зе щп эч гт бщ кэ ма ег

**Шифр Вижинера**



Г л у х о в а Д а р ь я В и т а л ь е в н а

З а щ и т а З а щ и т а З а щ и т а З а щ и

К л м ю б в з Д щ щ о я Й и л и ю ь м в ж и

**Расшифровка:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | иоарткдпвл\_натоа | Расшифровать с помощью магического квадрата   |  |  |  |  | | --- | --- | --- | --- | | 7 | 12 | 1 | 14 | | 2 | 13 | 8 | 11 | | 16 | 3 | 10 | 5 | | 9 | 6 | 15 | 4 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 7и | 12о | 1а | 14р |
| 2т | 13к | 8д | 11п |
| 16в | 3л | 10- | 5н |
| 9а | 6т | 15о | 4а |

Атлантида покров

**Код к практическому заданию на C#:**

|  |
| --- |
|  |
| using System;  using System.Text;  namespace OIB\_lab4  {  class Program  {  static void Main(string[] args)  {  var message = "глухова дарья витальевна";  Console.WriteLine(message);  var secretKey = 6;  Console.WriteLine(secretKey);  var keyWord = "защита";  Console.WriteLine(keyWord);  var encryptedText = CaesarEncrypt(message, secretKey);  Console.WriteLine("Цезарь: ");  Console.WriteLine(encryptedText);  Console.WriteLine("Виженер: ");  Vigenere(message, keyWord);  Console.WriteLine("Плейфер: ");  List<int> index = new List<int>();  string code = "глухова дарья витальевна";  string keyword = "защита";  string alphabet = "абвгдежзийклмнопрстуфхцчшщъыьэюя";  char[,] matrix = new char[4, 8];  keyword = new string(keyword.Distinct().ToArray());  // удаление букв ключевого слово из алфавита  for (int i = 0; i < keyword.Length; i++)  {  for (int j = 0; j < alphabet.Length; j++)  {  if (keyword[i] == alphabet[j]) alphabet = alphabet.Remove(j, 1);  }  }  // ввод данных в массив  Console.WriteLine("Матрица: ");  int k = 0, m = 0;  for (int i = 0; i < 4; i++)  {  for (int j = 0; j < 8; j++)  {  if (k < keyword.Length)  {  matrix[i, j] = keyword[k];  }  if (k >= keyword.Length)  {  matrix[i, j] = alphabet[m];  m++;  }  k++;  }  }  // вывод массива  for (int i = 0; i < 4; i++)  {  for (int j = 0; j < 8; j++)  {  Console.Write(matrix[i, j] + " ");  }  Console.WriteLine();  }  // нахождение индексов в матрице  for (int i = 0; i < code.Length; i++)  {  for (int j = 0; j < 4; j++)  {  for (int r = 0; r < 8; r++)  {  if (i < code.Length && code[i] == matrix[j, r])  {  index.Add(j);  index.Add(r);  }  }  }  }  Console.WriteLine("гл\tух\tов\tад\tар\tья\tви\tта\tль\tев\tна");  Encode(index, matrix);  }  private static string CaesarCodeEncode(string text, int k)  {  var fullAlphabet = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя";  var letterQty = fullAlphabet.Length;  var retVal = new StringBuilder();  for (int i = 0; i < text.Length; i++)  {  var c = text[i];  var index = fullAlphabet.IndexOf(c);  if (index < 0)  {  // Если символ не найден, то добавляем его в неизменном виде  retVal.Append(c);  }  else  {  var codeIndex = (letterQty + index + k) % letterQty;  retVal.Append(fullAlphabet[codeIndex]);  }  }  return retVal.ToString();  }  public static string CaesarEncrypt(string plainMessage, int key)  {  return CaesarCodeEncode(plainMessage, key);  }  public static string CaesarDecrypt(string encryptedMessage, int key)  {  return CaesarCodeEncode(encryptedMessage, -key);  }  static void Encode(List<int> index, char[,] matrix)  {  int a = 0;  for (int i = 0; i <= index.Count - 3; i += 4)  {  a = index[i + 3] + 1;  if (index[i] != index[i + 2] && index[i + 1] != index[i + 3])  {  Console.Write("{0,5}", matrix[index[i], index[i + 3]]);  Console.Write(matrix[index[i + 2], index[i + 1]]);  }  else if (index[i] == index[i + 2] && a >= 8)  {  a = 0;  Console.Write("{0,5}", matrix[index[i + 2], (index[i + 1] + 1) % 8]);  Console.Write(matrix[index[i], a]);  }  else if (index[i] == index[i + 2] && a < 8)  {  Console.Write("{0,5}", matrix[index[i + 2], (index[i + 1] + 1) % 8]);  Console.Write(matrix[index[i], a]);  }  else if (index[i + 1] == index[i + 3] && a >= 4)  {  a = 0;  Console.Write("{0,5}", matrix[a, index[i + 3]]);  Console.Write(matrix[(index[i] + 1) % 4, index[i + 1]]);  }  else if (index[i + 1] == index[i + 3] && a < 4)  {  Console.Write("{0,5}", matrix[a, index[i + 3]]);  Console.Write(matrix[(index[i] + 1) % 4, index[i + 1]]);  }  }  }  static void Vigenere(string message, string keyWord)  {  string m = message;  string k = keyWord;  int nomer; // Номер в алфавите  int d; // Смещение  string s; //Результат  int j, f; // Переменная для циклов  int t = 0; // Преременная для нумерации символов ключа.  char[] massage = m.ToCharArray(); // Превращаем сообщение в массив символов.  char[] key = k.ToCharArray(); // Превращаем ключ в массив символов.  char[] alfavit = { 'а', 'б', 'в', 'г', 'д', 'е', 'ё', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я' };  // Перебираем каждый символ сообщения  for (int i = 0; i < massage.Length; i++)  {  for (j = 0; j < alfavit.Length; j++)  {  if (massage[i] == alfavit[j])  {  break;  }  }  if (j != 33) // Если j равно 33, значит символ не из алфавита  {  nomer = j; // Индекс буквы  // Ключ закончился - начинаем сначала.  if (t > key.Length - 1) { t = 0; }  // Ищем индекс буквы ключа  for (f = 0; f < alfavit.Length; f++)  {  if (key[t] == alfavit[f])  {  break;  }  }  t++;  if (f != 33) // Если f равно 33, значит символ не из алфавита  {  d = nomer + f;  }  else  {  d = nomer;  }  // Проверяем, чтобы не вышли за пределы алфавита  if (d > 32)  {  d = d - 33;  }  massage[i] = alfavit[d]; // Меняем букву  }  }  s = new string(massage); // Собираем символы обратно в строку.  Console.WriteLine(s);  }  }  } |