Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Лабораторная работа № 5**

«Исследование криптографических шифров на основе перестановки символов»

Выполнил:

Студент: Герман А.Е.

ФИТ 3 курса 5 группы

Преподаватель: Савельева М.Г.

Минск 2024

# **1 Описание приложения**

Данное консольное приложение, разработанное на языке программирования C#, предназначено для разработки и использования приложений для реализации перестановочных шифров.

Основные функциональности приложения:

* Чтение введённых строк;
* Конвертация содержимого строк в матрицу;
* Маршрутная перестановка по спирали;
* Чтение ключевых строк в виде коллекции с парой: «буква» - «номер»;
* Множественная перестановка: вертикальная и горизонтальная.

# **2 Методика выполнения расчетов**

В данной лабораторной работе была поставлена цель создания приложения, позволяющее конвертировать текст и шифровать его. На листингах 2.1 и 2.2 представлен классы, реализующий данную функциональность.

|  |
| --- |
| class EncoderK  {  public string alphabet;  public int a, b;  public string editedAlphabet;  public EncoderK(string alphabet, int a, int b)  {  this.alphabet = alphabet;  this.a = a;  this.b = b;  this.editedAlphabet = editAlphabet(alphabet, a, b);  }  public string editAlphabet(string alphabet, int a, int b)  {  StringBuilder newAlphabet = new StringBuilder();  for (int iter = 0; iter < alphabet.Length; iter++)  {  int newPos = (a \* iter + b) % alphabet.Length;  newAlphabet.Append(alphabet[newPos]);  }  return newAlphabet.ToString();  }  public string encode(string text)  {  StringBuilder encodedText = new StringBuilder();  for (int iter = 0; iter < text.Length; iter++)  {  int pos = this.alphabet.IndexOf(text[iter]);  int newPos = (a \* pos + b) % alphabet.Length;  char encSymbol = this.editedAlphabet[newPos];  encodedText.Append(encSymbol);  }  return encodedText.ToString();  }  public string decode(string text)  {  StringBuilder decodedText = new StringBuilder();  int aInverse = ModInverse(a, alphabet.Length);  for (int iter = 0; iter < text.Length; iter++)  {  int pos = this.editedAlphabet.IndexOf(text[iter]);  int newPos = (aInverse \* (pos - b + alphabet.Length)) % alphabet.Length;  char decSymbol = this.alphabet[newPos];  decodedText.Append(decSymbol);  }  return decodedText.ToString();  }  private int ModInverse(int a, int m)  {  a = a % m;  for (int x = 1; x < m; x++)  {  if ((a \* x) % m == 1)  return x;  }  return 1;  }  } |

Листинг 2.1 – Класс **EncoderK**, реализующий функционал для работы аффинного шифрования Цезаря

|  |
| --- |
| public class EncoderPorta  {  private readonly Dictionary<char, int> letterToNumber;  private readonly Dictionary<int, char> numberToLetter;  public EncoderPorta()  {  letterToNumber = new Dictionary<char, int>();  numberToLetter = new Dictionary<int, char>();  string polishAlphabet = "AĄBCĆDEĘFGHIJKLŁMNŃOÓPRSŚTUWYZŹŻ";  for (int i = 0; i < polishAlphabet.Length; i++)  {  letterToNumber[polishAlphabet[i]] = i + 1;  numberToLetter[i + 1] = polishAlphabet[i];  }  }  public string Encrypt(string plaintext)  {  plaintext = plaintext.ToUpper();  List<int> encryptedNumbers = new List<int>();  foreach (char c in plaintext)  {  if (letterToNumber.ContainsKey(c))  {  int number = letterToNumber[c];  encryptedNumbers.Add(number);  }  }  return string.Join(" ", encryptedNumbers);  }  public string Decrypt(string ciphertext)  {  string[] numbers = ciphertext.Split(' ');  List<char> decryptedLetters = new List<char>();  foreach (string number in numbers)  {  if (int.TryParse(number, out int num))  {  if (numberToLetter.ContainsKey(num))  {  char letter = numberToLetter[num];  decryptedLetters.Add(letter);  }  }  }  return string.Join("", decryptedLetters);  } |

Листинг 2.2 – Класс **EncoderPorta**, реализующий функционал для шифрования Порта

# **3 Результаты работы приложения**

Для выполнения расчетов достаточно необходимо запустить приложение. Рисунки 3.1 - 3.2 показывают необходимые расчеты и вызовы методов, требуемые в данной лабораторной работе.

## **3.1 Маршрутная перестановка**

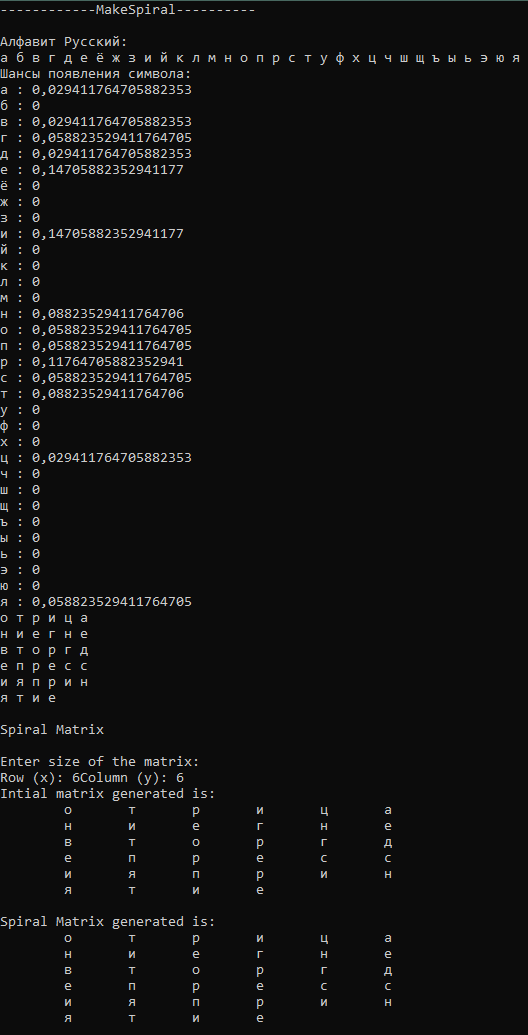
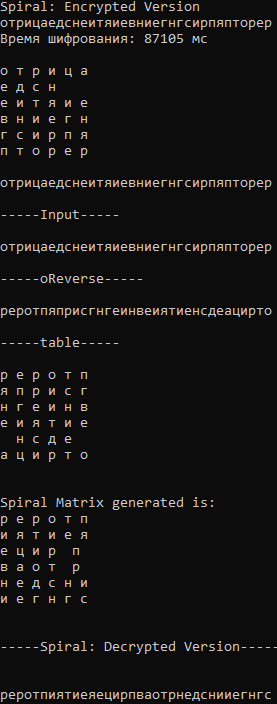
 

Рисунок 3.1 – Результат работы методов класса **EncoderK** с входного файла

В качестве тестового текста было использован файл russian1.txt. Используя маршрутную перестановку по спирали, была создана матрица размером 6x6, с помощью которого выполнялась перестановка. Время шифрования 87105 мс.

## **3.2 Множественная перестановка**

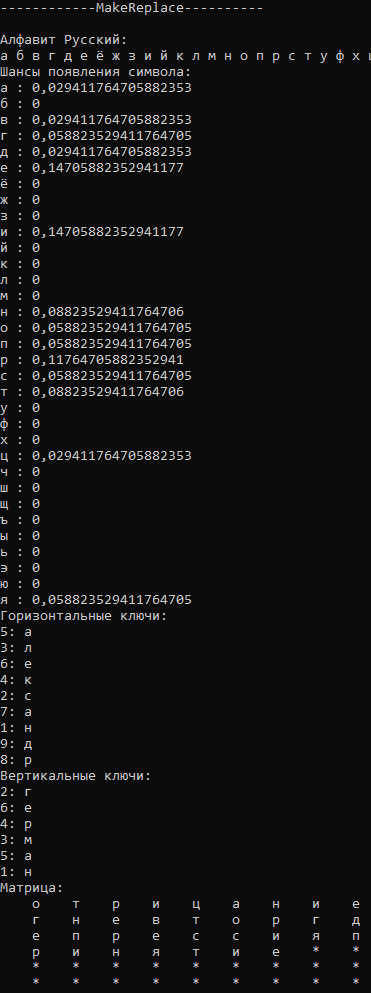
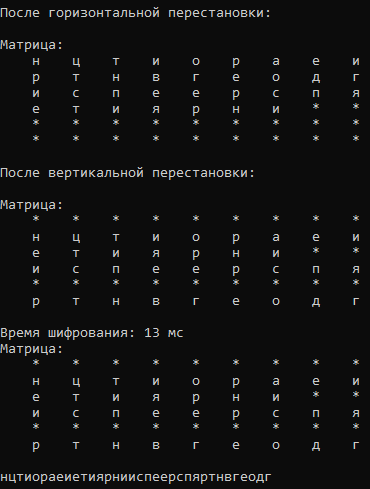
 

Рисунок 3.2 – Результат работы методов класса **EncodedPorta**

В качестве тестового текста было использован файл russian1.txt. Текст расшифровывается корректно, за исключением пробелов. Время шифрования 13 мс.

## **3.3 Гистограммы**

Рисунок 3.3 – Гистограмма вероятностей для исходного текста (Спиральная перестановка)

Рисунок 3.4 – Гистограмма вероятностей для исходного текста (Множественная перестановка)

## **3.4 График времени**

Рисунок 3.5 – График времени шифрования (Спиральная перестановка)

Рисунок 3.6 – График времени шифрования (Множественная перестановка)

Рисунок 3.7 – График времени шифрования (обе перестановки)

# **4 Вывод**

В ходе изучения теоретических материалов лабораторной работы и выполнения её практической части были изучены основы теории информации и анализа энтропии. Это консольное приложение, разработанное на языке программирования C#, предоставляет функциональности, направленные на измерение энтропии в различных сценариях, связанных с текстовыми данными. Гистограммы не претерпели изменений, так как не происходит каких-либо манипуляций c алфавитом, лишь с положением символов в строке.