

БУ ВО «Сургутский государственный университет»

Политехнический институт

Кафедра автоматизированных систем обработки информации и управления

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

ПО ДИСЦИПЛИНЕ «Теория Информации»

Выполнил: студент группы №606-12,

Речук Дмитрий Максимович

Принял: ст. преподаватель кафедры АСОИУ,

Гавриленко Анна Владимировна

Сургут 2025

Лабораторная

Введите предложение

На дворе - трава, на траве - дрова. Не руби дрова на траве двора !

Рассчитать

Загрузить текст

Буква	Вероятность	Шеннон-Фано	Хаффман
	0,2121	101	00
а	0,1515	110	110
р	0,1212	010	011
в	0,1061	0111	1111
н	0,0606	111	0101
е	0,0606	0011	1010
д	0,0606	100	1000
о	0,0606	0110	1001
т	0,0455	00011	0100
-	0,0303	0010	10110
.	0,0152	00010	101110
,	0,0152	000011	101111
у	0,0152	000010	111000
б	0,0152	000001	111001
и	0,0152	0000001	111010
!	0,0152	0000000	111011

Шеннон-Фано

$H = 3,48$
 $D = 0,04$
 $I = 3,64$

Хаффман

$H = 3,48$
 $D = 0,01$
 $I = 3,52$

```
using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Linq;
using System.Windows.Forms;

namespace Kod
{
    public partial class Form1 : Form
    {
        private string textVariable;
        private bool check = false;
        private string str;

        private const string ShannonColumn = "ShannonFano";
        private const string HuffmanColumn = "Huffman";
    }
}
```

```

public Form1()
{
    InitializeComponent();
    InitializeDataGridView();
}

private void InitializeDataGridView()
{
    resDataGridView.Columns.Add(ShannonColumn, "Шеннон-Фано");
    resDataGridView.Columns.Add(HuffmanColumn, "Хаффман");
}

private void Button1_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(TextBox.Text))
    {
        MessageBox.Show("Пожалуйста, введите текст или загрузите файл.",
"Неверный ввод");
        return;
    }

    try
    {
        if (textVariable != TextBox.Text)
            check = false;

        str = check ? textVariable.ToLower() : TextBox.Text.ToLower();

        // Calculate probabilities
        Dictionary<char, double> probabilities =
CalculateProbabilities(str);

        // Shannon-Fano
        var (shannonCodes, lFan) = CalculateShannonFano(probabilities);

        // Entropy
        double hFan = CalculateEntropy(probabilities.Values.ToArray());
        double dFan = lFan > 0 ? (lFan - hFan) / lFan : 0;

        // Huffman
        var (huffmanCodes, lHof) = CalculateHuffman(probabilities);
        double dHof = lHof > 0 ? (lHof - hFan) / lHof : 0;

        // Update UI
        PopulateDataGridView(probabilities, shannonCodes, huffmanCodes);

        ShIlabel.Text = $"l = {Math.Round(lFan, 2)}";
        ShHlabel.Text = $"H = {Math.Round(hFan, 2)}";
        ShDlabel.Text = $"D = {Math.Round(dFan, 2)}";

        HafIlabel.Text = $"l = {Math.Round(lHof, 2)}";
        HafHlabel.Text = $"H = {Math.Round(hFan, 2)}";
        HafDlabel.Text = $"D = {Math.Round(dHof, 2)}";
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Ошибка обработки данных: {ex.Message}",
"Ошибка");
    }
}

private void Button2_Click(object sender, EventArgs e)
{
    check = true;
    OpenFileDialog openFileDialog = new OpenFileDialog
    {

```

```

        Filter = "Текстовые файлы (*.txt)|*.txt|Все файлы (*.*)|*.*",
        Title = "Выберите текстовый файл"
    };

    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        try
        {
            string filePath = openFileDialog.FileName;
            textVariable = File.ReadAllText(filePath);
            TextBox.Text = textVariable;
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Ошибка чтения файла: {ex.Message}", "Ошибка
файла");
            check = false;
        }
    }

    private Dictionary<char, double> CalculateProbabilities(string text)
    {
        if (string.IsNullOrEmpty(text))
            return new Dictionary<char, double>();

        return text.GroupBy(c => c)
            .ToDictionary(
                g => g.Key,
                g => (double)g.Count() / text.Length
            );
    }

    private void PopulateDataGridView(Dictionary<char, double> probabilities,
string[] shannonCodes, List<string> huffmanCodes)
    {
        resDataGridView.Rows.Clear();
        var keys = probabilities.Keys.ToArray();
        for (int i = 0; i < keys.Length; i++)
        {
            resDataGridView.Rows.Add(
                keys[i],
                Math.Round(probabilities[keys[i]], 4),
                shannonCodes[i],
                huffmanCodes[i]
            );
        }
    }

    private double CalculateEntropy(double[] probabilities)
    {
        return probabilities.Sum(p => F(p));
    }

    private double F(double x)
    {
        return x > 0 ? -x * Math.Log(x, 2) : 0;
    }

    private (string[], double) CalculateShannonFano(Dictionary<char, double>
probabilities)
    {
        ShannonFanoEncoder encoder = new
ShannonFanoEncoder(probabilities.Values.ToArray());
        string[] codes = encoder.Encode();

        double avgLength = probabilities.Values

```

```

        .Zip(codes, (prob, code) => prob * code.Length)
        .Sum();

    return (codes, avgLength);
}

private (List<string>, double) CalculateHuffman(Dictionary<char, double>
probabilities)
{
    HuffmanTree huffmanTree = new HuffmanTree();
    huffmanTree.Build(probabilities);

    List<string> codes = huffmanTree.ReturnAlphabet();
    double avgLength = probabilities.Values
        .Zip(codes, (prob, code) => prob * code.Length)
        .Sum();

    return (codes, avgLength);
}
}

```