Московский государственный технический университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»


# Лабораторная работа № 5 по курсу

# «Базовые компоненты интернет-технологий»

Исполнитель: ИУ5-31, Черепанов Е.

Преподаватель: Гапанюк Ю.Е.

«___» _____

_____

Москва 2017г.

Условие задачи

Разработать программу, реализующую вычисление расстояния Левенштейна с использованием алгоритма Вагнера-Фишера.

1. Программа должна быть разработана в виде библиотеки классов на языке C#.

2. Использовать самый простой вариант алгоритма без оптимизации.

3. Дополнительно возможно реализовать вычисление расстояния Дамерау-Левенштейна (с учетом перестановок соседних символов).

4. Модифицировать предыдущую лабораторную работу, вместо поиска подстроки используется вычисление расстояния Левенштейна.

5. Предусмотреть отдельное поле ввода для максимального расстояния. Если расстояние Левенштейна между двумя строками больше максимального, то строки считаются несовпадающими и не выводятся в список результатов.

Код программы

```
using System; using System.Collections.Generic; using System.ComponentModel; using System.Data; using System.Drawing; using System.Linq; using System.Text; using System.Threading.Tasks; using System.Windows.Forms; using System.IO;

namespace лаб_5 {    public partial class Form1 : Form    {        public Form1()        { InitializeComponent();        }

        public List<string> SplitText(string fileName)        {            List<string> textByWords = new List<string>();            File.OpenRead(fileName);

            string text = File.ReadAllText(fileName);            string[] words = text.Split(' ', '.', ',', '!', '?', '(', ')', '=', '+', '-');            foreach (string temp in words)            {                if (!textByWords.Contains(temp)) {                    textByWords.Add(temp);                }            }            return textByWords;        }        private void button1_Click(object sender, EventArgs e)        {            openFileDialog1.Filter = "Текстовые файлы|*.txt";            openFileDialog1.ShowDialog();            label1.Text = openFileDialog1.FileName; }        private void openFileDialog1_FileOk(object sender, CancelEventArgs e)        {        } public void searchWords(string str, int maxDistance)        {            List<string> textBywords = SplitText(label1.Text);            int wordLen = str.Length;            String word = str.ToUpper(); foreach (string str1 in textBywords)            {                int tempLen = str1.Length;                int distance; if (wordLen == 0)                {                    distance = tempLen;                }                string temp = str1.ToUpper();                int[,] matrix = new int[wordLen + 1, tempLen + 1];                for (int i = 0; i <= wordLen; i++) matrix[i, 0] = i;                for (int j = 0; j <= tempLen; j++) matrix[0, j] = j;                for (int i = 1; i <= wordLen; i++)                {                    for (int j = 1; j <= tempLen; j++)                    { int symbEqual = (                        (word.Substring(i - 1, 1) ==                        temp.Substring(j - 1, 1)) ? 0 : 1);                        int ins = matrix[i, j - 1] + 1; //Добавление                        int del = matrix[i - 1, j] + 1; //Удаление                        int subst = matrix[i - 1, j - 1] + symbEqual;                        //Элемент матрицы вычисляется

                        //как минимальный из трех случаев                        matrix[i, j] = Math.Min(Math.Min(ins, del), subst);                        if ((i > 1) && (j > 1) && (word.Substring(i - 1, 1) == temp.Substring(j - 2,                        1)) &&                        (word.Substring(i - 2, 1) == temp.Substring(j - 1,                        1)))                        {                            matrix[i, j] = Math.Min(matrix[i, j], matrix[i - 2,                            j - 2] + symbEqual);                        }                    }                } if (matrix[wordLen, tempLen] <= maxDistance)                {                    listBox2.Items.Add(temp + " (" + matrix[wordLen, tempLen]                    + ")");                }            }        } private void button2_Click(object sender, EventArgs e)
```

```
{ listBox2.Items.Clear(); int distance; int.TryParse(textBox2.Text, out distance); searchWords(textBox1.Text, distance); }
    private void textBox1_TextChanged(object sender, EventArgs e) {
    }
    private void label1_Click(object sender, EventArgs e) {
    }
    private void listBox1_SelectedIndexChanged(object sender, EventArgs e) {
    }
    private void textBox1_TextChanged_1(object sender, EventArgs e)
    {
    }
    private void textBox2_TextChanged(object sender, EventArgs e) {
    }
    private void label2_Click(object sender, EventArgs e) {
    } } }
```

Проверка