МГТУ им. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5«Системы обработки информации и управления»

Дисциплина «Базовые компоненты интернет технологий»

# Отчёт по лабораторной работе №5

Выполнил:

Попов М.А.

ИУ5-35Б

Москва, 2019г

Задание:

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.

Текст программы:

***MainWindow.xaml.cs***

```
using System;
using System.IO;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
```

```
using System.Windows.Navigation;
using System.Windows.Shapes;
using Microsoft.Win32;
using System.Diagnostics;
using LLab5;

namespace Lab5_2
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        List< String> list = new List<String>();

        public MainWindow()
        {
            InitializeComponent();
        }

        private void Read_File_Button_Click(object sender,
RoutedEventArgs e)
        {
            OpenFileDialog first_dial = new OpenFileDialog();
            first_dial.Filter = "text_files|*.txt";

            if (first_dial.ShowDialog()== true)
            {
                Stopwatch mytimer = new Stopwatch();
                mytimer.Start();

                string text = File.ReadAllText(first_dial.FileName);

                char[] separators = new char[] { ' ', '.', ',', '!', '?', '/', '\t', '\n', '\r' };
                string[] textArray = text.Split(separators);
```

```
            foreach (string strTemp in textArray)
            {
                string str = strTemp.Trim();
            if (!list.Contains(str)) list.Add(str);
            }

            mytimer.Stop();
            this.textbox_for_timer.Text = mytimer.Elapsed.ToString();
            this.textbox_for_list.Text = list.Count.ToString();
        }
        else
        {
            MessageBox.Show("Выберите файл");
        }


    }

    private void Search_button_Click(object sender, RoutedEventArgs
e)
    {
        string word = this.Inputwords.Text.Trim();


        if (!string.IsNullOrWhiteSpace(word) && list.Count > 0 && word !=
"Введите слово для поиска")
        {

            string wordUpper = word.ToUpper();

            List<string> tList = new List<string>();
            Stopwatch t = new Stopwatch();
            t.Start();

            int maxRange = Int32.Parse(this.Max_range.Text.Trim());
```

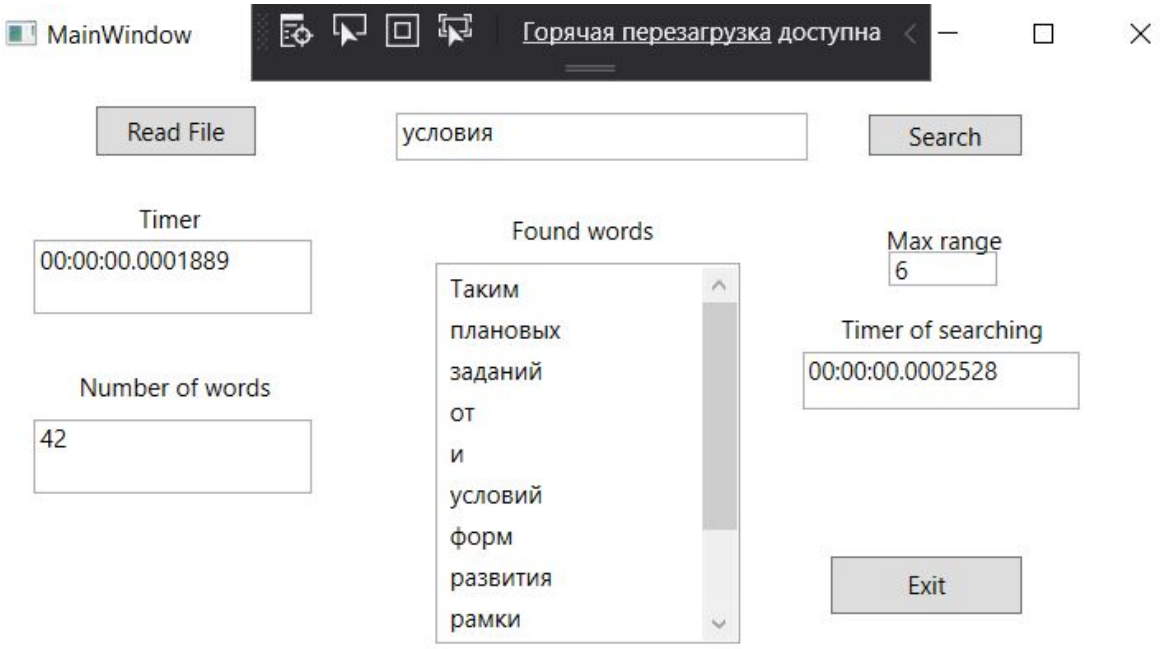
```
            foreach (string str in list)
            {
                if (DamLev.Distance(str, wordUpper) <= maxRange)
                {
                    tList.Add(str);
                }
            }
            t.Stop();
            this.Anothertimer.Text = t.Elapsed.ToString();

            this.found_words.Items.Clear();

            foreach (string str in tList)
            {
                this.found_words.Items.Add(str);
            }

        }
        else
        {
            MessageBox.Show("Видимо вы не выбрали файл");
        }
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        this.Close();
    }
  }
}
```

***Distance_Livenstein.cs***

```
using System;

namespace lab05
{
    public static class Distance_Levenstein

    {

        public static int Distance(string str1Param, string str2Param)
        {
            if ((str1Param == null) || (str2Param == null)) return -1; int str1Len
= str1Param.Length; int str2Len = str2Param.Length;
            if ((str1Len == 0) && (str2Len == 0)) return 0; if (str1Len == 0)
return str2Len; if (str2Len == 0) return str1Len;
            string str1 = str1Param.ToUpper();
            string str2 = str2Param.ToUpper();
            int[,] matrix = new int[str1Len + 1, str2Len + 1];
            for (int i = 0; i <= str1Len; i++)
                matrix[i, 0] = i;
            for (int j = 0; j <= str2Len; j++)
                matrix[0, j] = j;
            for (int i = 1; i <= str1Len; i++)
            {
                for (int j = 1; j <= str2Len; j++)
                {

                    int symbEqual = ((str1.Substring(i - 1, 1) == str2.Substring(j
- 1, 1)) ? 0 : 1);
                    int ins = matrix[i, j - 1] + 1;
                    int del = matrix[i - 1, j] + 1;
                    int subst = matrix[i - 1, j - 1] + symbEqual;
                    matrix[i, j] = Math.Min(Math.Min(ins, del), subst);
                    if ((i > 1) && (j > 1) && (str1.Substring(i - 1, 1) ==
str2.Substring(j - 2, 1)) && (str1.Substring(i - 2, 1) == str2.Substring(j - 1,
1)))
```

```
                {
                    matrix[i, j] = Math.Min(matrix[i, j],
                    matrix[i - 2, j - 2] + symbEqual);
                }
            }
        }
        return matrix[str1Len, str2Len];
    }
  }
}
```

Экранные формы с примерами выполнения программы:




Ссылка на репозиторий исходных кодов
GitHub: https://github.com/4Marvin2/Lab_C.git