МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н.Э.БАУМАНА

Факультет: «Информатика и системы управления»

Кафедра: «Системы обработки информации и управления (ИУ5)»

Отчет по лабораторной работе №5

По дисциплине «Базовые компоненты интернет-технологий»

Выполнил: Тураев Глеб Вадимович

Группа: ИУ5Ц-51Б

Преподаватель: Гапанюк Юрий Евгеньевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 25.12.2018

Подпись:

**Описание задания:**

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.

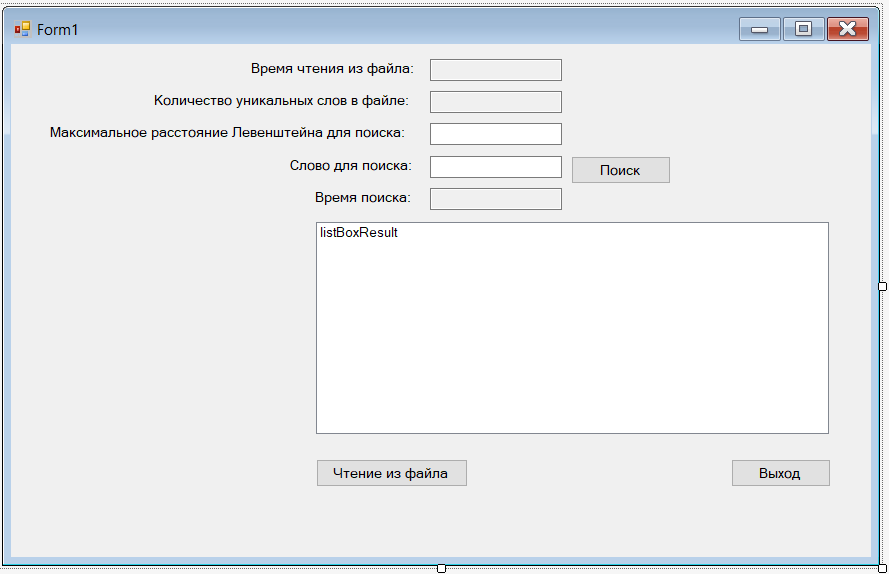
3. Дополнительно возможно реализовать вычисление расстояния Дамерау-Левенштейна (с учетом перестановок соседних символов).

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:**

*Form1.cs [Конструктор]*



*Form1.cs*

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.IO;

using System.Windows.Forms;

namespace LR5

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

List<string> list = new List<string>();

private static int Dist(string s1, string s2)

{

if (s1 == s2)

{

return 0;

}

var M = s1.Length + 1;

var N = s2.Length + 1;

var dist = new int[M, N];

for (var i = 0; i < M; i++)

{

dist[i, 0] = i;

}

for (var j = 0; j < N; j++)

{

dist[0, j] = j;

}

for (var i = 1; i < M; i++)

{

for (var j = 1; j < N; j++)

{

var diff = (s1[i - 1] == s2[j - 1]) ? 0 : 1;

dist[i, j] = Math.Min(

Math.Min(

dist[i - 1, j] + 1,

dist[i, j - 1] + 1

),

dist[i - 1, j - 1] + diff

);

}

}

return dist[M - 1, N - 1];

}

private static int DistDamerau(string s1, string s2)

{

if (s1 == s2)

{

return 0;

}

var M = s1.Length + 1;

var N = s2.Length + 1;

var dist = new int[M, N];

for (var i = 0; i < M; i++)

{

dist[i, 0] = i;

}

for (var j = 0; j < N; j++)

{

dist[0, j] = j;

}

for (var i = 1; i < M; i++)

{

for (var j = 1; j < N; j++)

{

if (s1[i - 1] == s2[j - 1])

{

dist[i, j] = dist[i - 1, j - 1];

}

var diff = (s1[i - 1] == s2[j - 1]) ? 0 : 1;

dist[i, j] = Math.Min(

Math.Min(

dist[i - 1, j] + 1,

dist[i, j - 1] + 1

),

dist[i - 1, j - 1] + diff

);

if (i > 1 && j > 1 && s1[i - 2] == s2[j - 1] && s1[i - 1] == s2[j - 2])

{

dist[i, j] = Math.Min(dist[i, j], dist[i - 2, j - 2] + 1);

}

}

}

return dist[M - 1, N - 1];

}

private void button1\_Click\_1(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog()

{

Multiselect = false,

Filter = "текстовые файлы|\*.txt"

};

var stopwatch = new Stopwatch();

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

stopwatch.Start();

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

char[] separators = new char[] { ' ', '.', ',', '!', '?', '/', '\t', '\n' };

foreach (var strTemp in text.Split(separators))

{

string str = strTemp.Trim();

if (!list.Contains(str))

{

list.Add(str);

}

}

stopwatch.Stop();

this.textBoxFileReadTime.Text = stopwatch.Elapsed.ToString() + " ms";

this.textBoxFileReadCount.Text = list.Count.ToString();

}

else

{

MessageBox.Show("Необходимо выбрать файл");

}

}

private void button3\_Click(object sender, EventArgs e)

{

var expectedSubstring = textBoxFind.Text.Trim();

if (!string.IsNullOrWhiteSpace(expectedSubstring) && list.Count > 0)

{

listBoxResult.Items.Clear();

var tempList = new List<string>();

var stopwatch = new Stopwatch();

var isFinded = false;

var maxDist = Int32.Parse(textBoxMaxDist.Text);

stopwatch.Start();

this.listBoxResult.BeginUpdate();

foreach (string word in list)

{

if (DistDamerau(word.ToUpper(), expectedSubstring.ToUpper()) <= maxDist)

{

listBoxResult.Items.Add(word);

isFinded = true;

}

}

this.listBoxResult.EndUpdate();

stopwatch.Stop();

if (!isFinded)

{

listBoxResult.Items.Add("No matches!");

}

this.textBoxExactTime.Text = stopwatch.Elapsed.ToString() + " ms";

}

}

private void button2\_Click(object sender, EventArgs e)

{

Close();

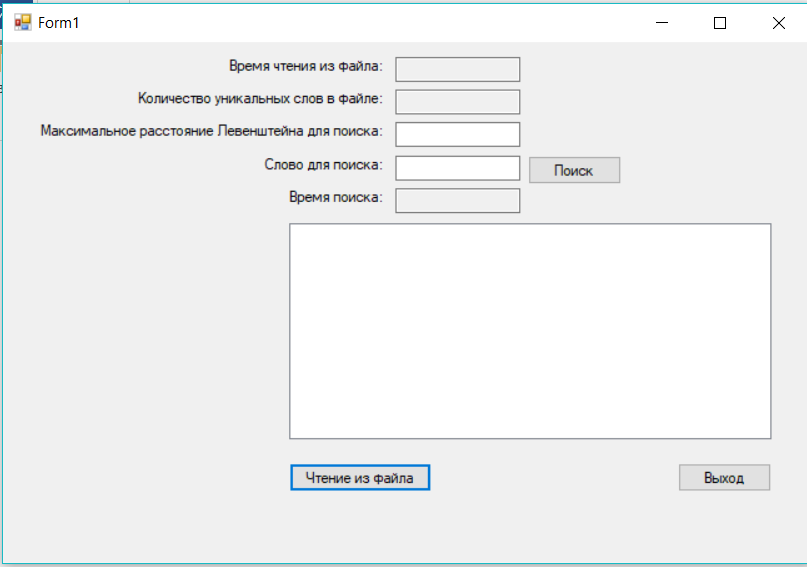
}

}

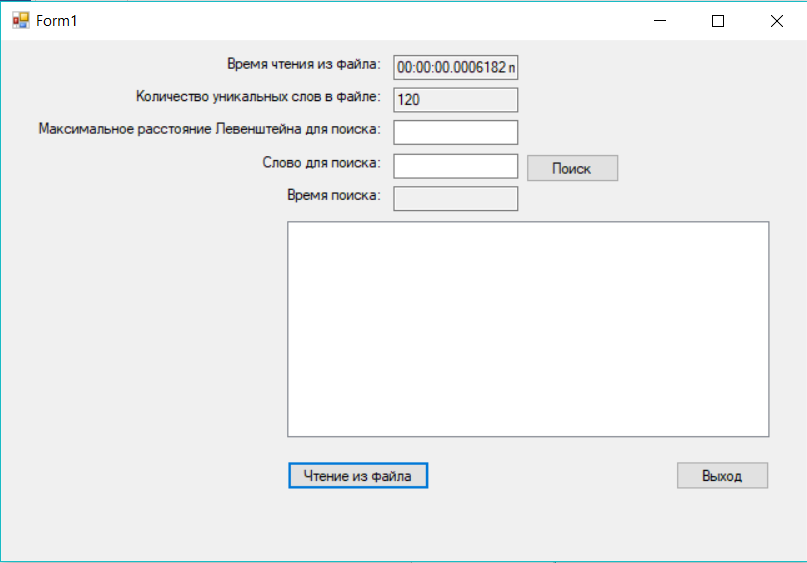
}

**Результат работы:**

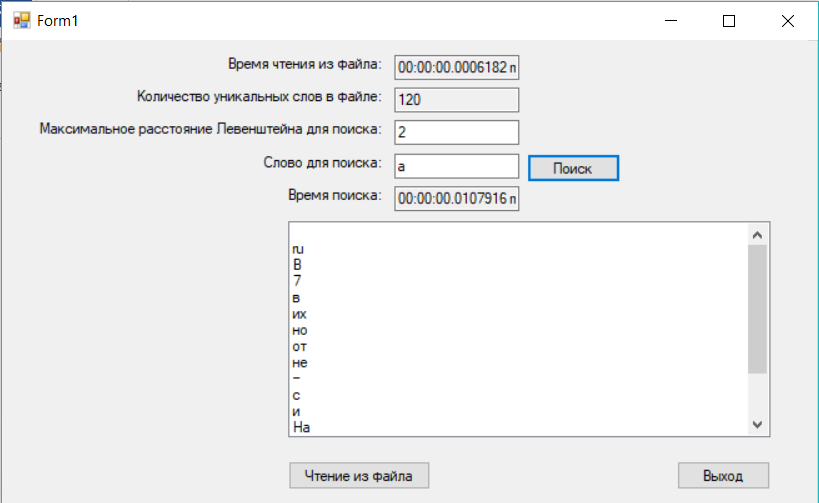
Запустим программу:



Нажмём на кнопку «Чтение из файла» и выберем текстовый документ:



Зададим «Максимальное расстояние Левенштейна для поиска:» и «Слово для поиска:»:



Затем, для выхода из программы, нажмём «Выход» и завершим работу.