МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИМ. Н. Э. БАУМАНА

Факультет "Информатика и системы управления"

### Кафедра ИУ5 “Системы обработки информации и управления”

Группа ИУ5-32Б

Отчёт по лабораторной работе №5

«Расстояния Левенштейна»

Студент: Преподаватель:

Ваганов Даниил Дмитриевич Гапанюк Юрий Евгеньевич

Дата: \_\_\_.\_\_\_.\_\_\_\_\_\_ Дата: \_\_\_.\_\_\_.\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_ Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020

**Условие 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:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Levenshtein\_distance

{

public class Class1

{

public static void WriteDistance(string str1Param, string str2Param, int maxsize)

{

int d = Distance(str1Param, str2Param);

if (d < maxsize)

Console.WriteLine("'" + str1Param + "' -> '" + str2Param + "' = " + d.ToString());

else

Console.WriteLine("Слова не совпадают");

}

public static int Distance(string str1, string str2)

{

if ((str1 == null) || (str2 == null)) { return -1; }

int str1Len = str1.Length;

int str2Len = str2.Length;

if ((str1Len == 0) && (str2Len == 0)) { return 0; }

if (str1Len == 0) { return str2Len; }

if (str2Len == 0) { return str1Len; }

string string1 = str1.ToUpper();

string string2 = str2.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];

}

}

}

**Анализ результатов:**

|  |  |
| --- | --- |
| *Тест* | *Вывод программы* |
| 1 |  |
| 2 |  |
| 3 |  |