Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5.

Курс «Базовые компоненты интернет-технологий»

Отчет по ЛР5

Выполнил:

студент группы ИУ5-31Б

Вардумян Арсен

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Юрий Евгеньевич

Подпись и дата:

г. Москва, 2020 г.

Постановка 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.

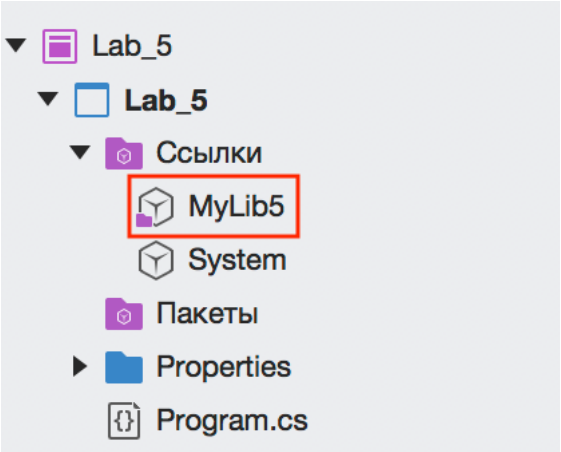
Модифицировать предыдущую лабораторную работу, вместо поиска подстроки используется вычисление расстояния Левенштейна.

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.

Разработка интерфейса класса

Класс Levenshtein реализован в виде библиотеки .dll:



Листинг программы

//Program.cs

```
using System;
using MyLib5;

namespace Lab_5
{
    class MainClass
    {
        public static void Main(string[] args)
        {
            Console.WriteLine(MyLib5.Levenshtein.LevenshteinDistance("пример",
"пирмер"));
        }
    }
}
```

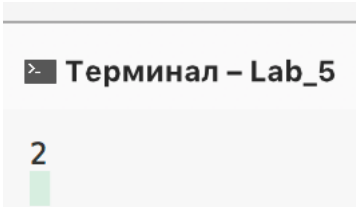
//Levenstein.cs

```
using System;
namespace MyLib5
{
    public static class Levenshtein
    {
```

```
        public static int LevenshteinDistance(string s1, string s2, bool damerau =
false)
        {
            if (s1 == null) throw new ArgumentNullException("s1 is null...");
            if (s2 == null) throw new ArgumentNullException("s2 is null...");

            if ((s1.Length == 0) && (s2.Length == 0)) return 0;
            if (s1.Length == 0) return s2.Length;
            if (s2.Length == 0) return s1.Length;

            s1 = s1.ToUpper();
            s2 = s2.ToUpper();

            int equal;
            int[,] matrix = new int[s1.Length + 1, s2.Length + 1];

            for (int i = 0; i <= s1.Length; i++) matrix[i, 0] = i;
            for (int j = 0; j <= s2.Length; j++) matrix[0, j] = j;

            for (int i = 1; i <= s1.Length; i++)
            {
                for (int j = 1; j <= s2.Length; j++)
                {
                    equal = (s1[i - 1] == s2[j - 1]) ? 0 : 1;
                    matrix[i, j] = Math.Min(Math.Min(matrix[i - 1, j] + 1,
matrix[i, j - 1] + 1), matrix[i - 1, j - 1] + equal);

                    if (damerau)
                    {
                        if ((i > 1) && (j > 1) && (s1[i - 1] == s2[j - 2]) &&
(s1[i - 2] == s2[j - 1]))
                        {
                            matrix[i, j] = Math.Min(matrix[i, j], matrix[i - 2, j
- 2] + equal);
                        }
                    }
                }
            }
            return matrix[s1.Length, s2.Length];
        }
    }

}
```

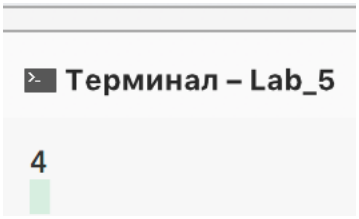
Анализ результатов

MyLib5.Levenshtein.LevenshteinDistance("пример", "пирмер");



MyLib5.Levenshtein.LevenshteinDistance("пример", "пирмер", damareu: true);

MyLib5.Levenshtein.LevenshteinDistance("4примеггр", "пирмер", damerau: true)



MyLib5.Levenshtein.LevenshteinDistance("4примеггр", "пирмер", damerau: false)