

Название: Вычисление расстояния Ливенштейна (Дамерлау-Ливенштейна).

Текст программы:

```
using System;
using System.Collections.Generic;

namespace LevenshteinLibrary
{
    public class StringDistance
    {
        public static int LD(string s1, string s2)
        {
            if (s1 == null || s2 == null)
                throw new ArgumentNullException("Strings cannot be null");

            int k = s1.Length;
            int n = s2.Length;
            int[,] d = new int[k + 1, n + 1];
            for (int i = 0; i <= k; i++) d[i, 0] = i;
            for (int j = 0; j <= n; j++) d[0, j] = j;
            for (int i = 1; i <= k; i++)
            {
                for (int j = 1; j <= n; j++)
```

```

        {
            int cost = s1[i - 1] == s2[j - 1] ? 0 : 1;
            d[i, j] = Math.Min(Math.Min(d[i - 1, j] + 1, d[i, j - 1] + 1),
d[i - 1, j - 1] + cost);
        }
    }
    return d[k, n];
}

```

```

public static int DLD(string s1, string s2)
{
    if (s1 == null || s2 == null)
        throw new ArgumentNullException("Strings cannot be null");

    int k = s1.Length;
    int n = s2.Length;
    int[,] d = new int[k + 1, n + 1];
    for (int i = 0; i <= k; i++) d[i, 0] = i;
    for (int j = 0; j <= n; j++) d[0, j] = j;
    for (int i = 1; i <= k; i++)
    {
        for (int j = 1; j <= n; j++)
        {
            int cost = s1[i - 1] == s2[j - 1] ? 0 : 1;
            d[i, j] = Math.Min(Math.Min(d[i - 1, j] + 1, d[i, j - 1] + 1),
d[i - 1, j - 1] + cost);
        }
    }
}

```

```

        if (i > 1 && j > 1 && s1[i - 1] == s2[j - 2] && s1[i - 2] ==
s2[j - 1])
        {
            d[i, j] = Math.Min(d[i, j], d[i - 2, j - 2] + cost);
        }
    }
}
return d[k, n];
}
}

```

```

public class LevenshteinFilter
{
    public static List<string> FilterStrings(string input, List<string>
candidates, int maxDist)
    {
        List<string> matches = new List<string>();
        foreach (string candidate in candidates)
        {
            int dist = StringDistance.LD(input, candidate); //Using the
short name here
            if (dist <= maxDist)
            {
                matches.Add(candidate);
            }
        }
        return matches;
    }
}

```

```
    }  
  }  
}
```

```
namespace LevenshteinApp
```

```
{
```

```
    using LevenshteinLibrary;
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

```
            Console.WriteLine("Введите строку для сравнения:");
```

```
            string input = Console.ReadLine();
```

```
            Console.WriteLine("Введите максимальное расстояние:");
```

```
            if (!int.TryParse(Console.ReadLine(), out int maxDist))
```

```
            {
```

```
                Console.WriteLine("Максимальное расстояние должно  
быть числом.");
```

```
                return;
```

```
            }
```

```
            Console.WriteLine("Введите строки для сравнения (через  
точку с запятой:");
```

```
            string[] candidates = Console.ReadLine()?.Split(';');
```

```
    if (candidates == null || candidates.Length == 0)
    {
        Console.WriteLine("Список строк не может быть  
пустым.");
        return;
    }
```

```
    List<string> matches = LevenshteinFilter.FilterStrings(input,  
new List<string>(candidates), maxDist);
```

```
    Console.WriteLine("Результаты:");
    if (matches.Count > 0)
    {
        Console.WriteLine("Совпадающие строки:");
        foreach (string s in matches)
        {
            Console.WriteLine($"- {s}");
        }
    }
    else
    {
        Console.WriteLine("Совпадающих строк нет.");
    }
}
```

Результаты работы программы:

Введите строку для сравнения:

rtyuiorwoiueyt

Введите максимальное расстояние:

2

Введите строки для сравнения (через точку с запятой):

ftyuioroiuyhgfж; fghjkl

Результаты:

Совпадающих строк нет.

Введите строку для сравнения:

йцукенгшщз

Введите максимальное расстояние:

цукенгшщ

Максимальное расстояние должно быть числом.

Введите строку для сравнения:

cat

Введите максимальное расстояние:

3

Введите строки для сравнения (через точку с запятой):

carry; catty

Результаты:

Совпадающие строки:

- carry

- catty