

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

ОТЧЕТ ПО ПРАКТИКЕ
Дистанция Левенштейна.
2 курс, группа 2ИВТ2(2)

Выполнил:

_____ Н. К. Скляр
«___» _____ 2024 г.

Руководитель:

_____ С. В. Теплоухов
«___» _____ 2024 г.

Майкоп, 2024 г.

Содержание

1. Теория	3
1.1. Техническое задание	3
1.2. Теоретическая часть	3
2. Ход работы	4
2.1. Код приложения	4
2.2. Работа программы	6

1. Теория

1.1. Техническое задание

Задание:

Измерить по модулю разность между двумя последовательностями символов.

1.2. Теоретическая часть

Расстояние Левенштейна (редакционное расстояние, дистанция редактирования) — метрика, измеряющая по модулю разность между двумя последовательностями символов. Она определяется как минимальное количество односимвольных операций (а именно вставки, удаления, замены), необходимых для превращения одной последовательности символов в другую. В общем случае, операциям, используемым в этом преобразовании, можно назначить разные цены. Широко используется в теории информации и компьютерной лингвистике.

Впервые задачу поставил в 1965 году советский математик Владимир Левенштейн при изучении последовательностей, впоследствии более общую задачу для произвольного алфавита связали с его именем.

Расстояние Левенштейна и его обобщения активно применяется:

- для исправления ошибок в слове (в поисковых системах, базах данных, при вводе текста, при автоматическом распознавании отсканированного текста или речи).

- для сравнения текстовых файлов утилитой diff и ей подобными. Здесь роль «символов» играют строки, а роль «строк» — файлы.

- в биоинформатике для сравнения генов, хромосом и белков.

С точки зрения приложений определение расстояния между словами или текстовыми полями по Левенштейну обладает следующими недостатками:

- При перестановке местами слов или частей слов получаются сравнительно большие расстояния;

- Расстояния между совершенно разными короткими словами оказываются небольшими, в то время как расстояния между очень похожими длинными словами оказываются значительными.

$$D(i, j) = \begin{cases} 0 & ; i = 0, j = 0 \\ i * deleteCost & ; j = 0, i > 0 \\ j * insertCost & ; i = 0, j > 0 \\ D(i - 1, j - 1) & ; S_1[i] = S_2[j] \\ \min(& \\ \quad D(i, j - 1) + insertCost & \\ \quad D(i - 1, j) + deleteCost & ; j > 0, i > 0, S_1[i] \neq S_2[j] \\ \quad D(i - 1, j - 1) + replaceCost & \\) & \end{cases}$$

$\min(a, b, c)$ возвращает наименьший из аргументов.

2. Ход работы

2.1. Код приложения

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;

int levenshteinDistance(const string& s1, const string& s2) {

    int m = s1.length();
    int n = s2.length();

    vector<vector<int>> dp(m + 1, vector<int>(n + 1, 0));

    for (int i = 0; i <= m; ++i) {
        dp[i][0] = i;
    }
    for (int j = 0; j <= n; ++j) {
        dp[0][j] = j;
    }
}
```

```

    for (int i = 1; i <= m; ++i) {
        for (int j = 1; j <= n; ++j) {
            if (s1[i - 1] == s2[j - 1]) {
                dp[i][j] = dp[i - 1][j - 1];
            }
            else {
                dp[i][j] = min({ dp[i - 1][j] + 1,
                                dp[i][j - 1] + 1,
                                dp[i - 1][j - 1] + 1 });
            }
        }
    }

    return dp[m][n];
}

int main() {
    setlocale(LC_ALL, "Rus");
    string s1, s2;

    cout << "Введите первое слово: ";
    cin >> s1;

    cout << "Введите второе слово: ";
    cin >> s2;

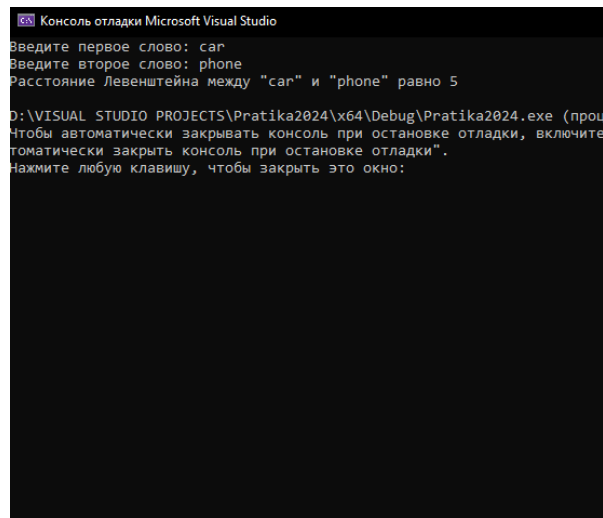
    int distance = levenshteinDistance(s1, s2);

    cout << "Расстояние Левенштейна между \"" << s1 << "\" и \"" << s2 << "\" равно: " << distance << endl;

    return 0;
}

```

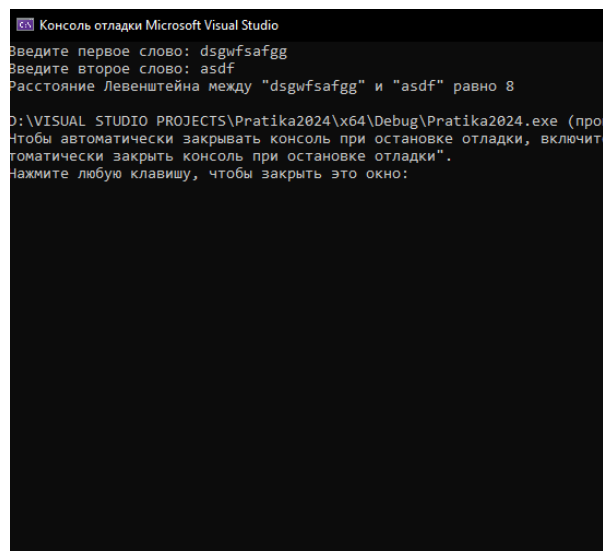
2.2. Работа программы



```
Консоль отладки Microsoft Visual Studio
Введите первое слово: car
Введите второе слово: phone
Расстояние Левенштейна между "car" и "phone" равно 5

D:\VISUAL STUDIO PROJECTS\Pratika2024\x64\Debug\Pratika2024.exe (проц
чтобы автоматически закрывать консоль при остановке отладки, включите
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рис.1 Пример работы программы.



```
Консоль отладки Microsoft Visual Studio
Введите первое слово: dsgwfsafgg
Введите второе слово: asdf
Расстояние Левенштейна между "dsgwfsafgg" и "asdf" равно 8

D:\VISUAL STUDIO PROJECTS\Pratika2024\x64\Debug\Pratika2024.exe (проц
чтобы автоматически закрывать консоль при остановке отладки, включите
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рис.2 Пример работы программы.