**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Построение и анализ алгоритмов»**

Тема: **Редакционное расстояние**

Вариант 4б.

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3388 |  | Беннер В.А. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2025

**Цель работы:**

Изучить алгоритмы Левенштейна для нахождения редукционного расстояния. Также реализовать задание по варианту.

**Задание.**

Расстоянием Левенштейна назовём минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.  
Разработайте программу, осуществляющую поиск расстояния Левенштейна между двумя строками.

Вариант 4б. Добавляется 4-я операция со своей стоимостью: замена двух символов на один символ.

**Реализация**

Программа реализует модифицированный алгоритм Левенштейна, который вычисляет редакционное расстояние между двумя строками с учетом дополнительной операции - замены двух символов на один. Этот алгоритм определяет минимальную стоимость преобразования строки S в строку T с использованием четырех операций редактирования.

**Шаги алгоритма**

**1. Инициализация:**

* **Создается матрица dp размером (m+1) x (n+1), где m и n - длины строк S и T.**
* **Первая строка матрицы заполняется стоимостью вставки всех символов T.**
* **Первый столбец заполняется стоимостью удаления всех символов S.**

**2. Заполнение матрицы:**

* **Для каждой позиции i в S и j в T:**
  + **Если символы S[i-1] и T[j-1] совпадают:**
    - **dp[i][j] = dp[i-1][j-1] (стоимость не меняется)**
  + **Если символы разные:**
    - **Рассчитываются стоимости всех возможных операций:**
      1. **Замена: dp[i-1][j-1] + replace\_cost**
      2. **Вставка: dp[i][j-1] + insert\_cost**
      3. **Удаление: dp[i-1][j] + delete\_cost**
      4. **Двойная замена (если i > 1): dp[i-2][j-1] + double\_replace\_cost**
    - **Выбирается операция с минимальной стоимостью.**

**3. Результат:**

* **Значение dp[m][n] содержит минимальную стоимость преобразования S в T.**

*Оценка сложности алгоритма:*

**Временная сложность:**

*O(m×n) - два вложенных цикла по длинам строк (m и n). На каждом шаге константное число операций (4 варианта редактирования).*

**Пространственная сложность**

O(m×n) при полной матрице.

**Тестирование**

Таблица 1. Тестирование.

|  |  |  |
| --- | --- | --- |
| Входные данные | Параметры операций | Результат |
| s: "kitten" t: "sitting" | replace=1 insert=1 delete=1 double=1 | 3 |
| s: "abcde" t: "abde" | replace=1 insert=1 delete=1 double=1 | 1 |
| s: "abcdef" t: "abzef" | replace=2 insert=1 delete=1 double=1 | 1 |
| s: "intention" t: "execution" | replace=1 insert=1 delete=1 double=1 | 5 |
| s: "a" t: "b" | replace=5 insert=3 delete=2 double=1 | 3 |
| s: "abcd" t: "acd" | replace=3 insert=2 delete=1 double=4 | 1 |

**Вывод**

В ходе лабораторной работы были написаны программы с использованием алгоритма Левенштейна.