**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: Динамическое программирование. Редакционное расстояние между строками.

|  |  |  |
| --- | --- | --- |
| Студент гр. 3388 |  | Павлов А.Р. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2025

**Цель работы:**

Изучить принципы динамического программирования, использовать их для решения задач на поиск редакционного расстояния разного вида между строками.

**Задание 1:**

Над строкой (будем считать строкой непрерывную последовательность из латинских букв) заданы следующие операции:

- replace(, a, b) – заменить символ a на символ b.

- insert(, a) – вставить в строку символ a (на любую позицию).

- delete(, b) – удалить из строки символ b.

Каждая операция может иметь некоторую цену выполнения (положительное число). Даны две строки A и B, а также три числа, отвечающие за цену каждой операции. Определите минимальную стоимость операций, которые необходимы для превращения строки A в строку B.

**Входные данные**: первая строка – три числа: цена операции replace, цена операции insert, цена операции delete; вторая строка – A; третья строка – B.

**Выходные данные**: одно число – минимальная стоимость операций.

**Sample Input**:

1 1 1

entrance

reenterable

**Sample Output**:

5

**Задание 2:**

Над строкой (будем считать строкой непрерывную последовательность из латинских букв) заданы следующие операции:

- replace(, a, b) – заменить символ a на символ b.

- insert(, a) – вставить в строку символ a (на любую позицию).

- delete(, b) – удалить из строки символ b.

Каждая операция может иметь некоторую цену выполнения (положительное число). Даны две строки A и B, а также три числа, отвечающие за цену каждой операции.

Определите последовательность операций (редакционное предписание) с минимальной стоимостью, которые необходимы для превращения строки A в строку B.

**Sample Input**:

Первая строка – три числа: цена операции replace, цена операции insert, цена операции delete; вторая строка – A; третья строка – B.

**Sample Output**:

Первая строка – последовательность операций (M – совпадение, ничего делать не надо; R – заменить символ на другой; I – вставить символ на текущую позицию; D – удалить символ из строки); вторая строка – исходная строка A; третья строка – исходная строка B.

**Задание 3:**

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.

**Пример:**

Для строк pedestal и stien расстояние Левенштейна равно 7:

Сначала нужно совершить четыре операции удаления символа: pedestal -> stal.

Затем необходимо заменить два последних символа: stal -> stie.

Потом нужно добавить символ в конец строки: stie -> stien.

**Реализация**

Используется динамическое программирование на матрице: хранит минимальную стоимость преобразования префикса в префикс .

**Шаги алгоритма**

Инициализируем первую строку матрицы стоимостью вставки символов, умноженной на количество символов; а первый столбец — ценой удаления, соответственно. Основной алгоритм заполняет матрицу слева направо: на каждом шаге выбирается минимум из стоимости трёх операций:  
1. Заменить символ ()

2. Вставить символ ()

3. Удалить символ ()

Согласно индвидуализации, стоимости считаются в зависимости от символов, стоящих на требуемых местах замены/удаления в строке B/A.

*Оценка сложности алгоритма:*

**Временная сложность**

Проход по всем ячейкам матрицы *dp.*

*Общая:*

**Сложность по памяти**

Матрица создаётся для всех префиксов входных данных.

*Итого*:

**Тестирование**

Таблица 1. Тестирование.

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 1 1  e  w  w 2  r 3 | 2 |
| 1 1 1  r  w  w 2  r 3 | 2 |
| 1 1 1  rq  w  w 2  r 3 | 3 |

**Вывод**

В ходе лабораторной работы были написаны программы с использованием динамического программирования. Также дополнительно был добавлен отладочный вывод для интерпретации результатов работы программы.