**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №5 (Вар. 1)**

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

Тема: Ахо-Корасик

|  |  |  |
| --- | --- | --- |
| Студент гр. 3388 |  | Сабалиров М.З. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2025

## Задание (Вариант 1. Выполнение на Stepik двух заданий в разделе 2)

Вход:

Первая строка содержит текст T (1< |T|< 100000).

Вторая строка содержит число n (1< n< 3000). Каждая следующая из n строк содержит шаблон из набора P = { p\_1, ldots, p\_n } (1< |p\_i|< 75).

Все строки содержат символы из алфавита { A, C, G, T, N }.

Выход:

Все вхождения образцов из P в T.

Каждое вхождение образца в текст представить в виде двух чисел - i p.

Где i - позиция в тексте (нумерация начинается с 1), с которой начинается вхождение образца с номером p (нумерация образцов начинается с 1).

Строки выхода должны быть отсортированы по возрастанию, сначала по номеру позиции, затем по номеру шаблона.

Задача:

Используя реализацию точного множественного поиска, решите задачу точного поиска для одного образца с джокером.

В шаблоне встречается специальный символ, именуемый джокером (wild card), который "совпадает" с любым символом. По заданному содержащему шаблоны образцу ( P ) необходимо найти все вхождения ( P ) в текст ( T ).

Например, образец ( ab??c?c ) с джокером ? встречается дважды в тексте \*zabuccbababcax\*.

Символ джокер не входит в алфавит, символы которого используются в ( T ). Каждый джокер соответствует одному символу, а не подстроке неопределённой длины. В шаблон входит хотя бы один символ не джокер, т.е. шаблоны вида ??? недопустимы. Все строки содержат символы из алфавита ({A, C, G, T, N}).

Вход:

- Текст ( T ) (( 1< |T|< 100000 ))

- Шаблон ( P ) (( 1< |P|< 40 ))

- Символ джокера

Выход:

- Строки с номерами позиций вхождений шаблона (каждая строка содержит только один номер).

- Номера должны выводиться в порядке возрастания.

## Выполнение работы

Для выполнения был использован алгоритм Ахо-Корасика. Алгоритм реализует поиск подстрок при помощи реализации конечного автомата на боре. В бор добавляются паттерны O(M), M — суммарная длина паттернов. Потом совершается итерация по тексту, на каждом шаге проверяется наличия вхождения патерна. Количество детей у каждой вершины бора не более k — длина алфавита. Значит вычисление всех хороших ссылок займет O(M\*k). А поиск в тексте займет O(N + t), N — длина текста, t — количество всех возможных вхождений паттернов в текст. Итого сложность O(M\*k + N + t). Память соответственно O(M\*k).

**Тестирование:**

|  |  |
| --- | --- |
| Input | Output |
| NTAG  3  TAGT  TAG  T | 2 2  2 3 |
| ABOBA  3  BA  BOB  MOEVM | 2 2  4 1 |

|  |  |
| --- | --- |
| Input | Output |
| ACTANCA  A$$A$  $ | 1 |
| ABOBA  B$ | 1  4 |

|  |  |
| --- | --- |
| Input | Output |
| ACTANCA  A$$A$  $  T |  |
| ACTANCAG  A$$A$  $  T | 4 |

**Выводы:**

В ходе работы был разработан и протестирован алгоритм для поиска вхождений шаблона с джокером, без джокера, с ограниченным джокером в тексте. Алгоритм использует автомат Ахо-Корасик для эффективного поиска подстрок, учитывая, что джокер может совпадать с любым символом. Были добавлены подробные отладочные выводы для отслеживания выполнения программы, что упрощает поиск и исправление ошибок. Программа корректно обрабатывает входные данные, находит все вхождения шаблона и выводит их в порядке возрастания. Решение успешно справляется с задачей, демонстрируя высокую производительность даже на больших объемах данных.