**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

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

Тема: **Алгоритм Кнута-Морриса-Пратта**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3388 |  | Дубровин Д.Н. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2025

**Цель работы.**

Разобраться с принципом работы алгоритма Кнута-Морриса-Пратта для поиска подстрок в строке. Использовать его для решения задач: поиска шаблона в тексте и проверки, является ли одна строка циклическим сдвигом другой.

## Задание.

Задача 1

Реализуйте алгоритм КМП и с его помощью для заданных шаблона P (|P| < 15000) и текста T (|T| < 5000000) найдите все вхождения P в T.

Вход:

Первая строка - P

Вторая строка - T

Выход:

Индексы начал вхождений P в T, разделенных запятой. Если P не входит в T, то вывести -1.

Задача 2

Заданы две строки A (|A| < 5000000) и B (|B| < 5000000).

Определить, является ли A циклическим сдвигом B (это значит, что A и B имеют одинаковую длину и A состоит из суффикса B, склеенного с префиксом B). Например, `defabc` является циклическим сдвигом `abcdef`.

Вход:

Первая строка - A

Вторая строка - B

Выход:

Если A является циклическим сдвигом B, индекс начала строки B в A, иначе вывести -1. Если возможно несколько сдвигов, вывести первый индекс.

## Выполнение работы

Алгоритм Кнута-Морриса-Пратта (KMP) эффективно находит все вхождения заданного шаблона в строку. Он использует префикс-функцию, позволяя избежать лишних сравнений символов и тем самым ускоряя поиск. Этот метод широко применяется в задачах обработки строк, когда требуется определить все позиции, с которых начинается подстрока P в тексте T.

**Реализация**

Для решения поставленной задачи были реализованы следующие функции:

* findPrefixFunction(pattern string) []int – вычисляет префикс-функцию для заданного шаблона, которая используется в алгоритме Кнута-Морриса-Пратта для эффективного поиска подстрок.
* FindPatternOccurrences(text, pattern string, firstOnly bool) []int – находит все вхождения шаблона в текст, используя KMP. Если firstOnly == true, возвращает только первое вхождение.
* IsCyclicShift(text, pattern string) int – проверяет, является ли pattern циклическим сдвигом text. Если да, возвращает индекс сдвига, иначе -1.

**Анализ сложности алгоритма**

Временная сложность:

* Вычисление префикс функции: O(m), где m — длина паттерна
* Поиск происходит за один проход по тексту длины n: сложность O(n)
* Итоговая сложность: O(m + n)

Пространственная сложность:

* Хранение массива префикс-функции pi: O(m), где m — длина паттерна
* Итоговая сложность: O(m)

**Тестирование:**

|  |  |
| --- | --- |
| Input | Output |
| ab  abab | 0,2 |
| ABOBA  BOB | -1 |
| BOB  ABOBA | 1 |

Таблица 1 — тестирование решения задания 1

|  |  |
| --- | --- |
| Input | Output |
| defabc  abcdef | 3 |
| ultsres  results | 4 |
| FKTI  MOEVM | -1 |

Таблица 2 — тестирование решения задания 2

**Выводы:**

В процессе работы был разработан и протестирован алгоритм поиска вхождений шаблона в тексте. Для оптимизации используется массив префиксных значений, что повышает эффективность поиска подстрок. В программу добавлены детализированные отладочные сообщения, упрощающие анализ работы и устранение ошибок. Алгоритм корректно обрабатывает входные данные, определяет все вхождения шаблона и выводит их в порядке возрастания. Решение показало высокую производительность, включая обработку больших объемов данных.