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

Студентка гр. 3388 _____ Басик В.В.

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) эффективный алгоритм, осуществляющий поиск подстроки в строке, используя то, что при возникновении несоответствия само слово содержит достаточно информации, чтобы определить, где может начаться следующее совпадение, минуя лишние проверки. 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:

- computePi(const string &s) → vector<int>

Вычисляет префиксную функцию для строки: массив максимальных длин совпадающих префиксов/суффиксов подстрок. Используется в KMP и поиске циклических сдвигов.

- kmpSearch(const string &P, const string &T) → vector<int>

Реализует алгоритм КМП для поиска всех вхождений шаблона P в тексте T. Возвращает список стартовых позиций совпадений.

- rotationSearch(const string &A, const string &B) → int

Определяет минимальный циклический сдвиг строки A для получения строки B. Возвращает -1 при невозможности преобразования через анализ удвоенной строки A+A.

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
bob baobobbib	3
BOB ABOBA	1

Таблица 1 — тестирование решения задания 1

Input	Output
gopher rehpog	-1
MIU MAU	-1

Таблица 2 — тестирование решения задания 2

Выводы:

В ходе разработки был реализован и проверен метод обнаружения совпадений образца в строке с применением оптимизационной техники. Для ускорения обработки используется предварительно рассчитанная таблица префиксных совпадений, которая сокращает количество избыточных вычислений. В программную реализацию интегрирован расширенный вывод диагностической информации для пошагового отслеживания логики выполнения. Система гарантированно находит все позиции совпадения паттерна, сортируя результаты по возрастанию индексов. Тестирование подтвердило устойчивую работу решения с различными типами входных данных, включая экстремально длинные текстовые последовательности, при сохранении стабильной производительности.