# Лабораторная работа № 4 по курсу дискретного анализа: Поиск за линейное время.

Выполнил студент группы 08-208 МАИ *Коростелев Дмитрий Васильевич.*

## Условие

1. Общая постановка задачи.

   Необходимо реализовать один из стандартных алгоритмов поиска образцов для указанного алфавита.

   Запрещается реализовывать алгоритмы на алфавитах меньшей размерности, чем указано в задании.

   **Формат входных данных** Искомый образец задаётся на первой строке входного файла.

   В случае, если в задании требуется найти несколько образцов, они задаются по одному на строку вплоть до пустой строки.

   Затем следует текст, состоящий из слов или чисел, в котором нужно найти заданные образцы.

   Никаких ограничений на длину строк, равно как и на количество слов или чисел в них, не накладывается.

   **Формат результата** В выходной файл нужно вывести информацию о всех вхождениях искомых образцов в обрабатываемый текст: по одному вхождению на строку.

   Для заданий, в которых требуется найти только один образец, следует вывести два числа через запятую: номер строки и номер слова в строке, с которого начинается найденный образец. В заданиях с большим количеством образцов, на каждое вхождение нужно вывести три числа через запятую: номер строки; номер слова в строке, с которого начинается найденный образец; порядковый номер образца.

   Нумерация начинается с единицы. Номер строки в тексте должен отсчитываться от его реального начала (то есть, без учёта строк, занятых образцами).

   Порядок следования вхождений образцов несущественен.

2. Вариант задания.

   (a) Номер варианта: 1-2

   (b) Вариант алгоритма: Поиск одного образца при помощи алгоритма Кнута-Морриса-Пратта.

   (c) Вариант алфавита: Числа в диапазоне от 0 до $2^{32} - 1$

## Метод решения

По заданию требуется реализовать алгоритм поиска подстроки в строке за линейное время, для сначала найден префикс функцию, которая будет для каждого элемента паттерна ставить в соответствие число - максимальную длину суффикса строки от первого до элемента, который равен префиксу всей строк. С помощью данной функции при КМП обходе строка патерна будет сдвигаться на число элементов заданное перфикс функцией.

Кроме этого, так как в здании сказано, что файл может быть сколь угодной длины, значит нужно научить программу обрабатывть поток частями, при этом не теряя и не перебирая несколько раз возможные вхождения.

## Описание программы

**std::vector<long long> prefix_function(std::vector<unsigned long int>& pattern)** - префикс-функция, принимает на вход паттерн, и затем строит по нему префикс-функцию. Итак, префикс-функция в нулевом элементе всегда равна 0. Префикс функция расширяется, слева направо проходится патерн, запоминаются совбодающие перфиксы и суфиксы, при этом, при расширении следующего значения префикс функции, учитываются сравнения, сделанные на предыдущем шаге.

**void KMP(std::vector<long long>& pf, std::vector<unsigned long int>& pattern, std::vector<unsigned long int>& text, int begin, std::vector<long long>& result)** - стандартный алгоритм КМП, принимает ранее расчитанную префикс функцию, паттерн, буффер с текстом, позицию, с которой начинать поиск и вектор в который будет записан результат.

**void PrintResults(std::vector<long long>& result, std::vector<long long>& lines, long long& lines_count, long long patter_size, long longstart_off)** - функция, которая выводит корректный результат на экран. Принимает вектор с индексами вхождения паттернов в буффер, вектор с каких индексов начинаюся новые строки в буффере, кол-во ранее считанных линий, размер занимаемой в начале части текста с прошлого вхождения, последняя выведенная позиция в прошлом вхождении. Данная функция позволяет синхронизировать результаты при поиске вхождений строк в текст если сам происк производится одинаковыми кусками (которые хранятся в буффере) или в целом тексте.

## Бенчмарк

Линейность поиска будем проверять на тестах разной размерности, при этом размер патерна фикисирован и равен 2. Для полноты проверки следовало бы еще изменять длину патрена, но даже простого взгляда на программу хватит, чтобы понять что скорость работы от длинны патерна зависит линейно, так как префикс функция считается

один раз за линейное время.

| № | **размер текста** | **Время** |
|---|---|---|
| 1 | 100 | 53 ms |
| 2 | 1000 | 57 ms |
| 3 | 2000 | 64 ms |
| 4 | 5000 | 100 ms |
| 5 | 10000 | 150 ms |
| 6 | 100000 | 447 ms |

## Дневник отладки

| **№** | **Вердикт** | **Проблема и решение** |
|---|---|---|
| 1-3 | CE | Исправление ошибок и предупреждений |
| 3-9 | WA | Ошибки в выводе результатов поиска |
| 10 | TL | Оптимизация программы |

## Недочёты

Функция вывода результата не работает, если размер буфера меньше чем в два раза, так как КМП при таком раскладе никогда не найдет вхождения в буффер, или результаты будут выведены некорректно.

## Выводы

КМП алгоритм довольно прост в реализации и использовании, большей трудностью для меня стал вывод информации на экран, приведение ее к читаемому виду. Стоит отметить, что не смотря на свою простоту с первого взгляда может показать, что данный алгоритм нелинеен. Однако при его детальном осмотре становится понятно, что при всех ситуациях, даже не смотря на вложенность циклов алгоритм имеет линейное число сравнений и вхождений относительно заданного текста или паттерна.