

# Лабораторная работа №5 по курсу дискретного анализа: Суффиксные деревья

Выполнил студент группы М8О-312Б-22 МАИ *Мамонтов Егор*.

## Условие

### Вариант: 2

Найти в заранее известном тексте поступающие на вход образцы с использованием суффиксного массива.

## Метод решения

Я предпринял решение задачи с помощью суффиксных массивов, то есть через "сортировку". Для начала я должен был написать функцию суффиксного массива, который вернет индексы, расположенные в особом порядке. Благодаря этому массиву суффиксных индексов я могу найти суффиксы. Далее я должен создать функцию, которая из первой строки, паттерна и массива суффиксных индексов вернет мне индекс вхождения паттерна в первую строку. В конце я должен буду сделать корректный вывод индексов в консоль.

## Описание программы

Для реализации алгоритма были реализованы следующие функции и структуры:

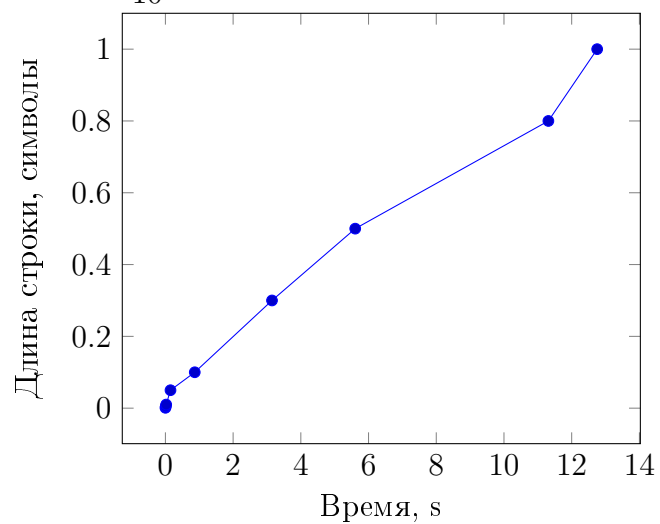
- `struct Symbol` - структура символа, содержащая в себе старый и новый эквивалент, индекс и символ.
- `std::vector<Symbol> suffixArray` - функция строит суффиксный массив для заданной первой строки и возвращает его в виде вектора `Symbol` с индексами.
- `std::string searchSubstr` - функция находит все вхождения подстроки `pattern` в первой строке.
- `bool sort_key` - функция используется для сортировки элементов `Symbol` в векторе `suff` по их `eq` значениям.

## **Дневник отладки**

1. Был получен WA на тесте №3. Для решения проблемы не нужно было выводить номер подстроки, которая не нашлась в первой строке.
2. Был получен WA на тесте №4. Для решения проблемы нужно было выводить до двоеточия именно номер подстроки, а не номер нахождения вхождения.

## Тест производительности

Алгоритм построения суффиксного массива работает за "суперлинейное" время, то есть сложность  $O(n \log n)$ , где  $n$  - длина первой строки.  
·10<sup>6</sup>



## Выводы

Я научился строить суффиксный массив, который является быстрым способом найти подстроку в строке, а также менее затратным по памяти, в отличие от суффиксного дерева Укконена.