**Алгоритмы на строках. Префикс и Z-функция**

**Префиксом строки** $S$ называется подстрока, начало которой совпадает с началом строки.

**Суффиксом строки** $S$ называется подстрока, конец которой совпадает с концом строки.

**Префикс-функция** строки $s[0...n-1]$ - это массив чисел $\pi[0...n-1]$, где в $\pi[i]$ хранится длина наибольшего суффикса подстроки $s[0...i]$, совпадающего с префиксом этой подстроки.

Например, для строки *abacaba* префикс-функция будет $\{0, 0, 1, 0, 1, 2, 3\}$.

**Z-функция** строки $s[0...n-1]$ - это массив чисел $Z[0...n-1]$, где в $Z[i]$ хранится длина наибольшей подстроки, начинающейся с $i$-того символа и совпадающей с началом строки $S$.

Например, для строки *abacaba* Z-функция будет $\{0, 0, 1, 0, 3, 0, 0\}$.

Обе функции можно посчитать за время $O(n)$ для строки длины $n$. Алгоритм для подсчета префикс-функции называется алгоритмом **Кнута-Морриса-Пратта**. Алгоритмы подсчета Z и префикс-функции обычно являются взаимозаменяющими. Для решения задач можно применять любой из них. Теперь разберём несколько задач, которые можно решить с помощью префикс-функции или z-функции.

**Поиск подстроки в строке** Пусть дано две строки $S$ и $T$ и надо найти все вхожения строки $S$ в строку $T$. Такую задачу можно решить за время $O(|S| + |T|)$. Для этого достаточно посчитать префикс или Z функцию от строки $S + '\$' + T$ и найти все позиции, где значение функции равно $|S|$ (для префикс функции придется из этих позиций вычесть длину строки $S$).

**Подсчет количества разных подстрок строки** Дана строка $S$. Необходимо посчитать количество разных подстрок строки $S$. Будем решать эту задачу итеративно. А именно, научимся, зная текущее количество различных подстрок, пересчитывать это количество при добавлении в конец одного символа.

Итак, пусть $k$ — текущее количество различных подстрок строки $s$, и мы добавляем в конец символ $c$. Очевидно, в результате могли появиться некоторые новые подстроки, оканчивавшиеся на этом новом символе $c$. А именно, добавляются в качестве новых те подстроки, оканчивающиеся на символе c и не встречавшиеся ранее.

Возьмём строку $t = s + c$ и инвертируем её (запишем символы в обратном порядке). Наша задача — посчитать, сколько у строки t таких префиксов, которые не встречаются в ней более нигде. Но если мы посчитаем для строки $t$ префикс-функцию и найдём её максимальное значение $\pi_{\max}$, то, очевидно, в строке t встречается (не в начале) её префикс длины $\pi_{\max}$, но не

большей длины. Понятно, префиксы меньшей длины уж точно встречаются в ней.

Итак, мы получили, что число новых подстрок, появляющихся при дописывании символа $c$, равно $s$.length() + 1 - $\pi_{\max}$.

Таким образом, для каждого дописываемого символа мы за O(n) можем пересчитать количество различных подстрок строки. Следовательно, за $O(n^2)$ мы можем найти количество различных подстрок для любой заданной строки.

Стоит заметить, что совершенно аналогично можно пересчитывать количество различных подстрок и при дописывании символа в начало, а также при удалении символа с конца или с начала.

Этот алгоритм можно переделать для работы Z функцией.

**Сжатие строки** Дана строка $s$ длины $n$. Требуется найти самое короткое её "сжатое"представление, т.е. найти такую строку $t$ наименьшей длины, что $s$ можно представить в виде конкатенации одной или нескольких копий $t$.

Посчитаем по строке $s$ префикс-функцию. Рассмотрим её последнее значение, т.е. $\pi[n-1]$, и введём обозначение $k = n-\pi[n-1]$. Если $n$ делится на $k$, то это $k$ и будет длиной ответа, иначе эффективного сжатия не существует, и ответ равен n.

Этот алгоритм можно переделать для работы Z функцией.