

Похожие имена

Автор задачи: Николай Будин, разработчик: Даниил Орешников

Первым действием стоит применить стандартный трюк, который используется, когда надо работать с циклическими сдвигами строки s — взять конкатенацию строки с самой собой (ss) и рассматривать все ее подстроки длины $|s|$. В частности, если нас интересует префикс некоторого циклического сдвига, то в строке ss нас просто интересуют любые подстроки длины не больше $|s|$.

С меньшими ограничениями эту задачу можно было бы решить следующим образом: перебрать циклический сдвиг самой короткой строки и применить префикс-функцию. С имеющимися ограничениями задачу можно было решить, применив хеши или суффиксный массив и алгоритм Касаи. Далее будет приведено решение с суффиксным массивом.

Давайте сконкатенируем вместе продублированные строки из ввода, разделив их специальным символом, не встречающимся в алфавите (например, '\$'). Заведем также массив `from`, отвечающий за номер исходной строки, которой соответствует каждый символ конечной строки, полученной в результате конкатенации.

Построим суффиксный массив. Заметим, что если ответ на задачу — a , то в полученной строке должны быть одинаковые подстроки длины a . Переформулируем это как «в строке есть суффиксы с общим префиксом длины a , с началами в каждой из исходных строк». Если применить алгоритм Касаи, мы сможем посчитать `lcp` — наибольший общий префикс соседних суффиксов в суффиксном массиве. Получается, что ответ — это такое a , что в массиве `lcp` есть отрезок лежащих подряд (потому что суффиксный массив лексикографически отсортирован) значений $\geq a$, соответствующие которым индексы из суфф. массива покрывают все множество исходных строк.

Для поиска максимального подходящего a можно воспользоваться двоичным поиском. Внутри цикла для конкретного значения a_0 достаточно пройти по массиву `lcp` и для каждого отрезка из значений не менее a_0 проверить, что значения `from` на нем покрывают все множество $1 \dots n$. Также стоит искусственно ограничить ответ сверху минимальной из длин исходных строк, чтобы избежать ситуации, когда трюк с конкатенацией увеличил потенциальный ответ.

Время работы решения — $\mathcal{O}(n \log n)$ на построение суффиксного массива и столько же на проход по `lcp` с бинпоиском.