

# Портальная пушка

Автор задачи и разработчик: Даниил Орешников

Сразу заметим, что третий запрос стандартно решается полиномиальным хэшированием. Если мы можем для каждой подстроки строки быстро узнавать ее хэш, то можем быстро сравнивать подстроки на равенство. Осталось быстро обрабатывать запросы первых двух типов.

Научимся быстро обновлять хэши для подстрок. Для этого будем отдельно обрабатывать каждый возможный символ от 'a' до 'z'. Заведем на каждый символ декартово дерево по явному ключу, которое будет хранить позиции вхождения этого символа. Помимо этого будем в каждой вершине ДД поддерживать сумму  $\text{prime}^i$  по всем позициям  $i$  в поддереве.

Чтобы посчитать хэш подстроки, будем итерироваться по всем символам и в каждом делать `split` соответствующего ДД, чтобы получить суммарный хэш позиций этого символа на отрезке. Хэш строки, соответственно, будет равен сумме  $s \cdot \text{hash}(c, l, r)$  по всем возможным символам  $s$ . Единственное отличие, что хэш будет умноженным на  $\text{prime}^l$ , поэтому при сравнении хэшей надо умножить хэш самой левой подстроки на  $\text{prime}^{|l_2 - l_1|}$ .

Чтобы отвечать на запросы первых двух типов,

1. для запроса первого типа просто перенесем соответствующий индекс  $i$  из дерева символа  $s_1$  в дерево символа  $s_2$ ,
2. для запроса второго типа перенесем все элементы меньшего дерева в большее.

Поскольку мы всегда переливаем меньшее дерево в большее, каждый элемент будет перенесен не более  $\log n$  раз, за исключением тех элементов, которые переносятся запросами первого типа. Таким образом, получим время работы  $\mathcal{O}(26 \cdot (n + q) \cdot \log^2 n)$ .