Биомаркеры

Автор задачи и разработчик: Константин Бац

Формально, в задаче требовалось найти максимальное кратное трем число m такое, что последовательность цифр числа m является подпоследовательностью цифр числа n.

Для решения первых двух подзадач, когда в числе n не больше шести значащих цифр, было достаточно просто перебрать все числа от 0 до n, кратные трем, и среди таких выбрать максимальное, у которого цифры являются подпоследовательностью цифр n. Проверку на то, что цифры одного числа являются подпоследовательностью цифр другого, можно осуществить за линейное от длин чисел время. Для этого достаточно перебирать цифры числа и для каждой находить минимальное следующее за предыдущей вхождение этой цифры в число n.

В третьей подзадаче k, количество цифр числа n, было ограничено сверху числом 18. Количество всех возможных подпоследовательностей цифр n равно $2^k < 10^6$, что позволяет перебрать такие подпоследовательности, проверить их на кратность трем и выбрать из них ту, которая дает максимальное число. Построить и сравнить два не более чем k-значных числа можно за время $\mathcal{O}(k)$, и время работы такого решения — $\mathcal{O}(k \cdot 2^k)$.

Для решения последующих подзадач требовалось свойство кратности целых чисел трем: *остаток от деления числа x на три равен остатку от деления суммы цифр числа x на три.*

Давайте посчитаем сумму цифр числа n, назовем ее s, а остаток от деления ее на три — r. Если s кратно трем, то есть $r = 0$, то ответ на задачу — число n. Иначе, рассмотрим случаи $r \neq 0$. Для простоты будем называть *противоположным остатком* к r число $2 - r$, то есть для одного — два, а для двух — один.

Если число n имеет остаток $r \neq 0$, то верно хотя бы одно из двух:

- в числе n есть не меньше одной цифры с остатком r;
- в числе n есть не меньше двух цифр с остатком, противоположным r.

Заметим, что если в числе n нет нулей, то чтобы сделать его кратным трем, потребуется удалить не больше двух цифр. Действительно, воспользуемся утверждением выше: если в числе есть цифра с остатком r, можно удалить только ее, иначе можно удалить две цифры с остатком, противоположным r остатком.

Ограничения четвертой подзадачи позволяли проверить оба случая, явно перебрать позиции удаляемых цифр, и выбрать максимальное число из тех, которые получаются после удаления. Заметим, что чем больше цифр мы удаляем, тем меньше число, поэтому удалять больше двух цифр из числа n не имеет смысла.

Для хранения числа n, промежуточных вычислений и сравнений можно использовать строки или массивы чисел. Сравнение двух чисел будет выполняться за $\mathcal{O}(k)$, а на перебор позиций уйдет $\mathcal{O}(k^2)$. Тогда время работы такого решения равно $\mathcal{O}(k^3)$.

Теперь рассмотрим более внимательно имеющиеся варианты. Пусть число n не содержит нулей, и из него можно удалить одну цифру, чтобы оно стало кратно трем. Тогда посмотрим на все позиции цифр, имеющих остаток r по модулю три.

Среди них могут быть такие $i \leqslant k - 1$, для которых следующая цифра на позиции $i + 1$ больше, чем текущая, то есть $n_i < n_{i+1}$. Докажем, что если удалить цифру с минимальной такой позиции i, то получится максимальное число, кратное трем, которое можно получить, удалив из n одну цифру. Действительно, при удалении цифры на позиции i мы получаем число, имеющее общий префикс с n длины $i - 1$, после которого следует цифра n_{i+1}.

Если обозначить за m число, образуемое префиксом числа n длины $k - 1$, то удаление цифры, большей, чем следующая за ней, приведет нас к получению числа, меньшего m. А удаление цифры на позиции i, меньшей, чем следующая за ней, позволит получить число, большее m уже в i-й цифре. Чем меньше будет такое i, тем больше будет конечное число.

Если же в записи числа n нет такой позиции $i \leqslant k - 1$, что остаток цифры на этой позиции равен r, и $n_i > n_{i+1}$, то выгодно удалить цифру, имеющую остаток r, находящуюся на максимальной из

всех таких цифр позиции. Это можно доказать аналогичными приведенным выше рассуждениями.

Если в числе n нет цифр с остатком r, то придется удалить две цифры противоположного остатка. С помощью доказанных выше утверждений можно свести задачу удаления двух таких цифр к удалению сначала одной такой цифры (либо находящейся как можно левее и меньшей своего правого соседа, либо находящейся как можно правее) по очереди.

Пятая подзадача выделялась тем, что цифры шли в порядке неубывания, что, с одной стороны, гарантировало отсутствие нулей, а с другой стороны, позволяло проще искать индексы интересных нам цифр. Действительно, если для любого i верно, что $n_i \leqslant n_{i+1}$, то достаточно взять минимальный индекс i с остатком r для цифры n_i. Если нам нужны две позиции цифр с противоположным остатком, то можно взять два таких минимальных индекса по очереди, и это даст правильный ответ.

Чтобы решить шестую подзадачу, нужно было полностью реализовать идею, описанную выше, не опираясь на неубывание цифр в числе. Время работы такого решения все еще будет равно $\mathcal{O}(k)$ — достаточно сделать константное количество полных проходов по числу. Заметим, что здесь мы сильно, хоть и неявно, опираемся на то, что n не содержит нулей. Однако комбинацией решения третьей и шестой подзадач уже можно было набрать 81 балл.

На самом деле, решение шестой подзадачи не верно для достаточно узкого класса чисел. Одно из таких чисел, например, равно 700222. Когда мы выбирали позиции, с которых нужно удалить цифры, мы не учли, что их удаление может породить ведущие нули. Кроме того, недостаточно применить алгоритм, описанный выше, и просто убрать ведущие нули, так как правильный ответ для такого числа равен 7002, а не 222.

Достаточно заметить, что ведущие нули такого рода в результате применения описанного алгоритма могут образоваться если число имеет вид

[`одна или две цифры, не кратные трем`] [`несколько нулей`] [`некоторый суффикс`].

Полное решение заключается в том, что нужно попробовать удалить нужное количество цифр не только из всего числа, но и отдельно только из его суффикса. Затем среди полученных результатов можно выбрать максимальный путем сравнения за $\mathcal{O}(k)$. Перед сравнением требуется избавится от возможного префикса из ведущих нулей.

Таким образом, несколько запусков алгоритма поиска и удаления одной цифры из числа будут работать за $\mathcal{O}(k)$, как и потенциальное сравнение нескольких возможных результатов, а значит и общее время работы равно $\mathcal{O}(k)$.