Разбор задачи «Как покормить дракона»

Переведем оба данных нам момента времени в минуты: если от начала суток прошло h часов и m минут, то всего прошло $60h + m$ минут.

Пусть в первый день от начала дня до кормления прошло t_1 минут, а во второй — t_2 минут. Тогда между этими двумя моментами прошло сначала $24 \cdot 60 - t_1$ минут до конца первого дня и еще t_2 минут от начала второго дня. Значит, всего прошло $\Delta t = 24 \cdot 60 - t_1 + t_2$ минут.

Осталось только перевести время из минут в правильный формат. Для этого достаточно заметить, что Δt минут — это $\lfloor \frac{\Delta t}{60} \rfloor$ часов и $\Delta t \ mod \ 60$ минут, где $a \ mod \ b$ — это остаток от деления числа a на число b.

Разбор задачи «Секрет Драконьего глаза»

Для решения первой подзадачи достаточно перебрать все пары подотрезков и проверить все указанные в задаче условия. Такое решение будет работать за $O(|s|^5)$.

Для решения второй подзадачи решение можно соптимизировать следующими способами:

- достаточно перебирать только подотрезки одинаковой длины
- считать сумму на каждом отрезке можно с помощью префиксных сумм

Таким образом, для решения второй подзадачи сначала следует перебрать общую длину отрезков, затем левые концы обоих отрезков, а после этого проверить все указанные в задаче условия, считая сумму на каждом отрезке с помощью предподсчитанных $p_i = \sum_{j=1}^{j=i} s_i$. Такое решение будет работать за $O(|s|^3)$.

Для решения третьей подзадачи можно еще больше соптимизировать решение: перебрав общую длину отрезков l, сгруппируем все отрезки длиной l по значению суммы на них. После этого, перебрав левую границу одного отрезка, мы можем легко узнать, есть ли еще один отрезок длиной l с такой же суммой. Таким образом, получаем решение за $O(|s|^2)$.

Однако, чтобы решить задачу на полный балл, нужно использовать другое решение. Для начала посмотрим на символы по краям строки — s_1 и s_n. Если они одинаковые, то мы сразу получаем ответ на задачу: подотрезки $[1, n-1]$ и $[2, n]$ удовлетворяют всем условиям задачи и имеют максимально возможную длину. Теперь рассмотрим случай, когда эти символы различны. Предположим, что $s_1 = 0$, а $s_n = 1$.

Найдем минимальный индекс i, такой, что $s_i = 1$, а также максимальный индекс j, такой, что $s_j = 0$ (то есть все символы перед s_i равны 0, а все символы после s_j равны 1). Утверждение: ответ на задачу — либо пара отрезков $[1, j-1]$ и $[2, j]$, либо пара отрезков $[i, n-1]$ и $[i+1, n]$. Почему?

Во-первых, несложно заметить, что обе эти пары отрезков удовлетворяют всем условиям задачи. Во-вторых, предположим, что существует ответ лучше — пара отрезков $[l_1, r_1]$ и $[l_2, r_2]$ с большей длиной, удовлетворяющая условию. Тогда несложно доказать, что $l_1, l_2 \leqslant i$, а также $r_1, r_2 \geqslant j$, что означает, что у обоих отрезков есть общая часть $[i, j]$. Оставшиеся же части отрезков состоят либо из одних нулей ($[l_1, i)$ и $[l_2, i)$), либо из одних единиц ($(j, r_1]$ и $(j, r_2]$). А из того, что и суммы, и длины у отрезков должны быть одинаковые, несложно доказать, что тогда $l_1 = l_2$ и $r_1 = r_2$, что противоречит условию о том, что отрезки должны быть разными.

Таким образом, оптимальный ответ — одна из двух пар отрезков $[1, j-1]$ и $[2, j]$, или $[i, n-1]$ и $[i+1, n]$. Суммарная асимптотика решения — $O(|s|)$.

Разбор задачи «Деревня викингов»

Если есть цикл, конечно, ответа не существует.

Иначе, рассмотрим произвольную вершину. Посмотрим на ребра, в нее ведущие. Рассмотрим среди них самую правую в топологической сортировке. Если она не является ее предком в дереве (которое требуется построить), то у нее в поддереве не будет этой вершины (так как остальные вершины находятся левее). А значит, у каждой вершины, кроме корня, предком является вершина, ведущая в нее, находящаяся как можно правее в топологической сортировке. Таким образом можно построить дерево.

Как проверить, что это дерево подходит? Достаточно проверить, что для любого ребра ориентированного графа, вершина, из которой исходит ребро, является предком второй вершины. Это можно сделать с помощью времен входа-выхода.

Таким образом, время работы есть $O(n + m)$.

Разбор задачи «Путешествие по островам»

Заметим, что если для каждой пары островов мы научимся находить минимальное расстояние, которое нужно пролететь Беззубику, чтобы попасть с одного острова на другой, то после этого нужно будет найти минимальное расстояние в графе от одной вершины до другой, где вершины — это острова. Это стандартная задача, которая решается, например, алгоритмом Дейкстры.

Находить минимальное расстояние между двумя многоугольниками можно за разную асимптотику. Минимальный по длине путь между двумя многоугольниками — это всегда отрезок, соединяющий две стороны данных многоугольников. Поэтому, можно для каждой пары многоугольников перебрать все пары сторон и сделать два вложенных тернарных поиска, получится асимптотика $O(k^2 \cdot P^2)$ для каждой пары многоугольников, где P — количество итераций тернарного поиска, необходимых, чтобы достичь нужной точности ($P \approx 50$).

Далее, можно заметить, что раз никакие два многоугольника не пересекаются, не пересекаются и никакие две стороны двух разных многоугольников. Поэтому всегда найдется отрезок, соединяющий вершину одного многоугольника и сторону другого, который будет минимальным по длине путем между ними. Значит, можно избавиться от вложенного тернарного поиска. Получается асимптотика $O(k^2 \cdot P)$.

Далее, можно заметить, что для нахождения расстояния от точки до отрезка, не нужно использовать тернарный поиск, и его можно вычислить за $O(1)$, тогда асимптотика становится равна $O(k^2)$ для каждой пары многоугольников.

Наконец, полное решение использует алгоритм вычисления растояния между двумя многоугольниками с помощью суммы Минковского. Построение суммы Минковского двух многоугольников можно выполнить или за $O(k \cdot \log(k))$, или за $O(k)$. Подробнее про нахождение расстояния между двумя выпуклыми многоугольниками с помощью суммы Минковского можно почитать здесь: `http://neerc.ifmo.ru/school/io/archive/20190302/geometry_doc.pdf`.