Разбор задачи «Аккаунты»

Построим бор по всем $2 \times n$ строкам. Теперь решаем следующую задачу: разделить строки, заканчивающиеся в вершинах, на такие пары, что в каждой паре вершина одной строки является предком вершины другой (в таком случае первая будет префиксом второй).

Обходим наш бор с помощью обхода в глубину, начиная из вершины, соответствующей пустой строке. Попадая в вершину, добавляем номера всех строк, которые в ней заканчиваются, в стек. Затем запускаем обход в глубину по очереди от каждого ребенка вершины.

После этого будем смотреть на последний элемент стека. Если он равен номеру какой-то строки, заканчивающейся в текущей вершине, значит мы не сопоставили эту строку кому-то из строк в вершинах-потомках текущей вершины. Значит, единственное, что остается — сопоставить эту строку либо с такой же неиспользованной строкой из текущей вершины, если такая имеется, либо с какой-то еще не использованной строкой из вершины-предка, то есть второму сверху элементу стека (все неиспользованные вершины лежат в стеке в порядке входа в них, значит вершина второй строки сверху будет нестрогим предком текущей). Удаляем два верхних элемента и повторяем процедуру со следующим элементом на вершине стека.

Асимптотика: $O(L)$, где L - суммарная длина всех строк.

Разбор задачи «AliKingspress»

Для начала решим задачу с использованием $O(x \cdot n)$ памяти, а затем придумаем, как это решение соптимизировать.

Посчитаем стандартную динамику $dp_{x,n}$ — минимальное количество дней, требуемое для получения x бонусов, учитывая то, что сейчас мы заходили n дней подряд. Переходы в этой динамике довольно логичные и описаны в условии: $dp_{x,n} \to dp_{x-a_n,\min(n+1,|a|-1)}$ и $dp_{x,n} \to dp_{x-a_0,1}$ (во втором случае мы считаем, что после пропуска дня нет смысла ждать еще несколько дней, поэтому сразу авторизируемся в следующий день)

Однако это решение использует $O(x \cdot n)$ памяти, что при данных ограничениях превышает 400 мегабайт и не укладывается в лимит по памяти. Чтобы устранить это, применим стандартный прием оптимизации памяти в динамике. Заметим, что из состояния (x, n) мы переходим только в те состояния (x', n'), где $x - x' <= 10^3$ (ограничение на элемент массива a). Тогда мы можем хранить не все x строк в матрице dp, а только последние 1000 из них, это соптимизирует нашу память в 1000 раз и позволит уложиться в лимит по памяти.

Разбор задачи «Протокол «Судного дня»»

Для начала, посчитаем для каждой вершины, в какой вершине с наименьшим расстоянием до магазина можно оказаться, проехав по не более чем одному тоннелю. Это наилучший из следующих вариантов: ответы для детей этой вершины, сама вершина, и все концы тоннелей, начинающихся в этой вершине.

Теперь пользуясь подсчитанным значением для проезда по одному тоннелю из вершины, найдем в какой вершине с наименьшим расстоянием до магазина можно оказаться, проехав из вершины i по не более чем 2^j тоннелям. Пересчитывается следующим образом: $d[i][2^j] = d[d[i][2^{j-1}]][2^{j-1}]$.

Чтобы обработать запрос, надо разложить k из запроса по степеням двойки и совершить скачки такого размера.

Разбор задачи «Плохая многозадачность»

Заменим каждое число a_i в массиве на $a'_i = \lfloor (a_i + b - 1)/b \rfloor$ — это количество секунд, которое требуется, чтобы i-я программа завершилась. Теперь можем считать, что каждая программа за секунду выполняет 1 операцию.

Все программы выполняются по циклу: $1 \to 2 \to 3 \to \ldots \to n \to 1 \to \ldots$. Количество этих циклов будет равно $a'_0 - 1$, после этого программа уничтожения запустится еще один раз и завершится. Несложно видеть, что тогда ответ равен $a'_0 + \sum_{i=1}^{i=n} \min(a'_i, a'_0 - 1)$.

Разбор задачи «Разбиение на пары»

Для начала рассмотрим тех агентов, которые хотят работать со своими коллегами из того же агенства. Объединим их между собой столько сколько возможно. При этом если агентов с такими

предпочтениями нечетное количество, то их мы пока что отложим в недовольных. Очевидно, что иметь пары, где оба агента довольны выгодно, так как в случае другого разбиения кто-то из них может быть не доволен и ответ не улучшиться.

Затем рассмотрим тех, кто хочет работать с другим агентством. Пусть Кингсманов из них больше чем Стейтсманов (обратный вариант рассматривается аналогично) и сформируем из них наибольшее возможное количество пар. После объединения в остатке получим какое-то количество агентов Кингсман. Кому-то из них можно дать в пару остаток от агентов из раньше рассмотренных предпочтений, таким образом сформируем до двух пар, где ровно один доволен.

Остальные, к сожалению, будут иметь дискомфорт. Останется только вывести их число.

Разбор задачи «Гарри и носки»

Мысленно положим левые носки в ряд так, что все носки лежат непрерывным и занумеруем их в таком порядке.

По сути мы хотим найти число таких перестановок правых носков, что ни для одной позиции цевого и правого носка не совпадают.

Пусть $A\{x_1, x_2, \ldots, x_k\}$ - множество перестановок правых носков, про которые верно, что носки с номерами $\{x_1, x_2, \ldots, x_k\}$ будут иметь пару такого же цвета.

В таком случае ответом на задачу является $|A\{\emptyset\}/\bigcup(A\{x_1\}, A\{x_2\}, \ldots, A\{x_n\})|$. Научимся считать $|A\{x_1, x_2, \ldots, x_k\}|$. Пусть в множестве x_i p_1 носков цвета 1, p_2 носков цвета $2, \ldots, p_m$ носков цвета m. Тогда мы выберем, какая будет пара для этих носков, а у остальных может быть любая пара. Способов так сделать: $\binom{c_1}{p_1} \cdot \binom{c_2}{p_2} \cdot \ldots \cdot \binom{c_m}{p_m} \cdot \binom{n-k}{c_1-p_1} \cdot \binom{n-k-(c_1-p_1)}{c_2-p_2} \ldots$.

По формуле включений-исключений $\bigcup(A\{x_1\}, A\{x_2\}, \ldots, A\{x_n\}) = \sum(-1)^k \cdot |A\{x_1, x_2, \ldots, x_k\}|$. Давайте для каждого k от 0 до n посчитаем сумму размеров множеств A с таким k. Для этого посчитаем $dp_{color,used}$ - число способов найти пару носкам первых $color$ цветов так, чтобы $used$ носков имели пару свокго цвета. Переход — это перебор числа носков, среди носков данного цвета, которые будут иметь пару своего цвета.

Более формально: $dp_{i,j} = \sum_{k=0}^{\min(j,cnt_i)} dp_{i-1,j-k} \cdot \binom{cnt_i}{k}^2 \cdot k!$, а ответ равен $\sum_{i=0}^{n} dp_{n,i} \cdot (n-i)! \cdot (-1)^i$.

Суммарное время на подсчёт $dp_{i,j}$ — $O(n^2)$.

Разбор задачи «Плеер Кингсманов»

В данной задаче была дана перестановка. Нужно было проверить, правда ли, что данная перестановка является начальной (1, 2, 3, …, n) в точности до циклического сдвига. Если это не так, то вывести первую позицию, в которой это становится понятно. Все, что нужно сделать — это проверить, для любой пары соседних элементов $(a_i + 1) \mod n = a_{i+1}$, предварительно уменьшив все a_i на 1. Как только нашли невыполнение данного условия — выводим ответ. Иначе, если условие выполнилось для всех пар, значит перестановка является начальной в точности до циклического сдвига.

Разбор задачи «Башни»

Посмотрим на самую левую, самую верхнюю, самую нижнею и самую правую башни. Заметим, что они точно лежат в покрытой области и являются самыми крайними точками покрытой области. Теперь построим границу нужной области, которая находится между самой левой и самой верней клеткой. Отсортируем все клетки по x. Тогда если мы сначала встретили точку (x1, y1), а потом точку (x2, y2) где y2 > y1, то мы добавляем в нашу границу два отрезка [(x1, y1), (x2, y1)] и [(x2, y1), (x2, y2)]. После чего мы повторяем тоже самое для всех остальных частей границы нужной области. В конце не забываем закрасить область внутри.

Крайним случаем является ситуация, где все башни лежат на одной диагонали. И представленный алгоритм на нем найдет не минимальное по площади покрытие. Поэтому этот случай нужно рассмотреть отдельно.