Задача A. Наряди елку

Давайте отсортируем все новогодние игрушки по диаметру. Тогда заметим, что в качества оптимального ответа мы всегда берём какой-то подотрезок игрушек в данной сортировке — если это не так, то мы можем добрать любые игрушки между самой левой и самой правой в нашем текущем ответе, и улучшить его. Значит, достаточно перебрать игрушку с минимальным диаметром, и двигать указатель на игрушку с максимальным диаметром, пока разница их диаметров не превосходит K, и обновить оптимальный ответ всеми игрушками между ними. Если каждый раз начинать с очередной игрушки, то сложность решения получится $O(N^2)$.

Заметим, что указатель на правую границу ответа нужно оставлять там же, где он оказался на прошлой итерации цикла по левым границам. Тогда оба указателя движутся только вправо, и мы на отсортированном массиве решаем задачу методом двух указателей за линейное время. Тогда вычислительную основную сложность в решении задачи вносит сортировка. Сложность такого решения составляет $O(NlogN)$.

Задача B. Участки леса

Давайте предподсчитаем "префиксные"суммы количеств деревьев в прямоугольниках, левый верхний угол которых находится в квадрате (1,1) а правый нижний — в (i, j), чтобы за $O(1)$ узнавать количество деревьев в любой подматрице. Делается это методом включений-исключений с учетом наличия дерева в квадрате $a_{i,j}$:

$P_{i,j} = P_{i-1,j} + P_{i,j-1} - P_{i-1,j-1} + (a_{i,j} == 1)$.

Теперь, для решения задачи за $O(N^2 \cdot M^2)$ достаточно перебрать два угла дома, проверить, что его площадь хотя бы S, и что деревьев на его территории не больше K. Такое решение набирало 65 баллов.

Для решения на полный балл нужно перебирать только подходящие прямоугольники.

Переберём левый нижний угол дома. Будем поддерживать его текущую максимально возможную высоту, и будем постепенно увеличивать его ширину. Для очередной ширины, нам необходимо уменьшать высоту, по сравнению с предыдущей шириной, пока территория дома содержит $> K$ деревьев. Чтобы посчитать ответ, нам достаточно по формуле определить минимальную необходимую высоту дома для текущей ширины, чтобы площадь была хотя бы S, и прибавить $max(0, MaxWidth - MinWidth + 1)$ к ответу. Такое решение работает за $O(N \cdot M \cdot (N + M))$ и набирает 100 баллов.

Задача C. Бутфол

В начале рассмотрим следующую задачу: пусть у нас есть набор из N предметов с весами до S, и мы хотим проверить, какие суммы можно набрать каким-то подмножеством этих предметов. Эта задача решается с помощью динамического программирования за время $O(N^2 \cdot S)$: пусть $dp_{i,j}$ - можем ли мы набрать сумму j, используя первые i предметов, $j \leqslant i \cdot S$. Это решение можно ускорить до времени $O(N \cdot N \cdot S/64)$, используя $std :: bitset$.

Тогда решение исходной задачи может выглядеть так: в начале проверим с помощью нашей динамики, что все люди без Тимы могут разбиться на команды, если это не так, то ответ пуст. После этого давайте перебирать людей, пересчитывать динамику без них, перебирать возможные силы Тимы до $N * S$, и проверять что остальные люди могут разбиться на команду силы $(totalSum - a_i + TimPower)/2$. Если хотя бы для одного человека, не участвующего в игре, какая-та сила Тимы не подходила, то она не может находиться в итоговом ответе. Таким образом, мы получили решение за $O(N^2 \cdot S + N^3 \cdot S)$, которое набирало от 28 баллов. Если добавить в решение $bitset$, то получим асимптотику $O(N^2 \cdot S + N^3 \cdot S/64)$, что могло пройти ещё какие-то группы.

Для решения задачи на полный балл потребуется следующий трюк: давайте в нашей динамике считать вместо того, возможно ли набрать какую-то сумму, количество способов это сделать. Тогда добавление предмета от удаления предмета отличается не сильно: достаточно пройтись циклом for и вычесть значения динамики в правильном порядке: $dp[i + s] -= dp[i]$ для всех i от 0 до $N \cdot S$, или же $dp[i + s] += dp[i]$ для всех i от $N \cdot S$ до 0. Таким образом, мы можем пересчитать динамику без какого-то человека за время $O(N \cdot S)$, и откатить ее назад за это же время. Так как количество способов может быть экспоненциально большим, можно его считать по какому-нибудь большому простому модулю(например $10^9 + 7$). Получаем решение за $O(N^2 \cdot S)$, которое набирало 100 баллов.

Задача D. НЛО

Решение в лоб работает за $O(K \cdot max(N, M))$: просто поддерживаем текущие высоты башен, и проходом по массиву на каждый запрос находим первые R башен нужной высоты, а затем уменьшаем их значения на 1. Оно набирало от 20 до 30 баллов, в зависимости от эффективности реализации.

Для решения задачи на больший балл нужны какие-то структуры данных. Нам нужно уметь обновлять значения в двумерном массиве, и находить первые $R \leqslant 10$ башен высоты хотя бы h в какой-то строке или столбце. Для этого можно поддерживать структуру данных на каждой строке и каждом столбце, которая умеет быстро изменять значение и находить первый элемент больше данного, например, дерево отрезков на максимум. Наивно можно делать бинпоиск с запросом максимума на отрезке за $O(log^2 N)$, но на 100 баллов это не должно было заходить :). Первый элемент больше данного правее какой-то позиции можно искать в ДО с помощью спуска за $O(logN)$. Тогда итоговое решение работает за $O(K \cdot R \cdot logN)$ и набирает 100 баллов.