Разбор задачи «Расследование убийства»

Если в формулу для $\beta(n,k)$ подставить формулу для $\beta(n-1,k)$, получится следующее:

$$\beta(n,k) = \frac{n-1+k}{n} \cdot \beta(n-1,k)$$

Раскрывая формулу выше рекурсивно, получается:

$$\beta(n,k) = \frac{(n-1+k)\cdot(n-1+k-1)\cdot\ldots\cdot k}{n\cdot(n-1)\cdot\ldots\cdot 2\cdot 1} \cdot \beta(0,k) = \frac{(n-1+k)!}{n!\cdot(k-1)!}$$

Предподсчитая заранее значения $n!$ и обратных к ним по модулю 998244353, можно узнавать значения $\beta(n,k)$ за $O(1)$.

Разбор задачи «Бюджет»

Решим задачу с помощью динамического программирования. Будем поддерживать массив $dp[i][j]$, в ячейке которого будем хранить значение «true», если на i-ом префиксе можно получить сумму j, «false» — если нельзя. База динамики: $dp[0][f[i]] = true$ (если $f[i]$ попадает в отрезок $[a,b]$), $dp[0][-f[i]] = true$ (аналогично, если $-f[i]$ попадает в отрезок $[a,b]$). Соответственно, во внешнем цикле будем перебирать все префиксы (по i), во внутреннем — суммы от a до b (по j). Если на предыдущем префиксе набралась сумма j, то есть $dp[i-1][j] == true$, значит на текущем префиксе набирается сумма $j + f[i]$, а также $j - f[i]$.

Если на последнем префиксе (то есть на всем массиве) набралась хотя бы одна сумма, лежащая на отрезке $[a,b]$, значит ответ на задачу существует. Чтобы восстановить ответ, необходимо знать, прибавляли ли мы или вычитали текущее число в массиве, будем также хранить эти данные в ячейке массива dp. Тогда, зная, какая сумма могла быть получена на последнем префиксе и какой знак был у последнего числа в массиве при получении этой суммы, мы можем переходить по суммам на меньших префиксах и получать знаки остальных чисел.

Асимптотика: $O(n \cdot (b-a))$.

Разбор задачи «Проникновение в реликварий»

Пусть в исходной последовательности 1 имела индекс i. Тогда что бы проверить, возможно ли это, надо для каждого j от 1 до n проверить, что $|a_{(i+j-1) \bmod n} - j| \leqslant 1$. Заметим, что для каждого элемента это условие выполнится не более трёх раз, поэтому весь алгоритм будет работать за линейное время.

Разбор задачи «Шум»

Для начала заметим, что чтобы восстановить какой-то вид исходного массива нужно к данному применить такую же функцию шума. Затем отсортируем массив по возрастанию, и при рассмотрении каждого элемента будем пытаться получить из него наименьший возможный такой, что в мы его еще не получали.

Пусть x текущий элемент, а y, предыдущее значение, тогда возможны три случая:

- $x - r > y$, тогда предположим, что элемент x был равен $x - r$ в первоначальном массиве.
- $x + r < y$, тогда все числа из диапазона $[x-r; x+r]$, уже были получены на предыдущих элементах, а значит, элемент x никак не привести к уникальному.
- иначе можем предположить, что вместо x раньше было значение $y + 1$, поставим его в ответ.

Соответственно, ответ — это сумма первых и третьих случаев, однако можно было сначала поставить числа таким способом, а потом уже посчитать ответ.

Для доказательства оптимальности данного подхода рассмотрим два числа x_0 и x_1, такие что $x_0 < x_1$. Очевидно, что пересечение диапазонов $[x_0 - r; x_0 + r]$ и $[x_1 - r; x_1 + r]$ не выгодно занимать при рассмотрении x_0, так как мы уменьшим количество возможных вариантов для x_1. В случае же, если мы возьмем минимальное число из $[x_0 - r; x_0 + r]$, которое до этого не брали, мы оставим наибольшее возможное количество вариантов для x_1.

Сложность решения $O(n\log(n)+n)$.

Разбор задачи «Садоводство в поезде»

Заметим, что сажать растения выгоднее всего в порядке убывания времен роста — так растения, которым расти нужно дольше всего будут расти, пока мы сажаем другие.

Отсортируем исходный массив a по убыванию, и посадим растения в таком порядке. Ответом будет $\max_{i=0..n-1}(a_i+i)$.

Разбор задачи «Безопасный пароль»

Заменим каждый третий повторяющийся символ на самый редко встречаемый в строке, не равный соседним. Очевидно, что количество вхождений таких символов никогда не станет больше половины длины. Подсчитаем, сколько раз встречается каждый из символов в строке. Пусть нашелся символ, который встретился чаще, чем в половине позиций. Тогда заменим минимальное количество вхождений этого символа на самые редко встречаемые в строке, не равный соседним.

Разбор задачи «Стабильность транзакций»

Для начала заметим интересный факт, для двух транзакций $a_i < a_j$, нет смысла разбивать транзакцию a_i, но не разбивать транзакцию a_j, так как максимум в такой ситуации точно не измениться, а минимум может уменьшится, то есть ухудшить ответ.

Второй полезный факт: разбивать всегда надо на две равные части по тем же самым соображениям.

Тогда отсортируем транзакции по возрастанию и переберем, начиная с какой позиции мы будем бить транзакции на две. Пусть первая транзакция, которую разбили имеет номер k. Тогда:

- $MIN = min(a_1, a_k/2)$
- $MAX = max(a_{k-1}, a_n/2)$

Переберем все k, найдем оптимальный ответ. Сложность решения $O(n\log(n)+n)$.

Разбор задачи «Вагоны Восточного Экспресса»

Будем решать данную задачу с помощью перебора. Давайте зафиксируем, сколько раз проводник будет освобождать вагон целиком. Пусть это число равно i. В таком случае, нужно освободить i вагонов так, чтобы максимальное количество людей в одном из оставшихся вагонов было минимально (пусть это количество равно x). Тога проводнику потребуется ровно x выпускать по одному пассажиру из вагонов, где еще остались люди. Следовательно, нужно минимизировать сумму $x+i$. Поступим следующим образом — отсортируем вагоны по количеству людей в них в невозрастающем порядке. Тогда, зафиксировав какое-то i мы будем целиком освобождать первые i вагонов, а x будет равно a_{i+1} или 0, если i было равно n. Итого, взяв минимум по всем i от 0 до n среди значений $i+a_{i+1}$ мы получим оптимальный ответ.