Разбор задачи «Морской бой»

Посчитаем минимальное количество клеток поля, необходимое, чтобы расположить корабли, в зависимости от числа k. Поскольку количество кораблей равняется $1 + 2 + \ldots + k$, количество клеток, разделяющих корабли, равняется этой сумме, вычисляемой по известной формуле $\frac{k \cdot (k+1)}{2} - 1$. Единицу отнимаем потому что разделяющих клеток на одну меньше, чем кораблей.

Теперь посчитаем количество клеток, занимаемое непосредственно кораблями. Это количество по условию равняется следующей сумме: $1 \cdot k + 2 \cdot (k-1) + 3 \cdot (k-2) + \ldots + k \cdot 1$. Раскрыв скобки и вынеся k, а также минус, за скобки, получим следующее выражение: $(1+2+\ldots+k) \cdot k - (1 \cdot 2 + 2 \cdot 3 + \ldots + (k-1) \cdot k)$. Заменив скобки эквивалентными формулами, получим: $\frac{k \cdot (k+1)}{2} \cdot k - \frac{(k-1) \cdot k \cdot (k+1)}{3}$. Заметим, что функция $f(k) = \frac{k \cdot (k+1)}{2} - 1 + \frac{k \cdot (k+1)}{2} \cdot k - \frac{(k-1) \cdot k \cdot (k+1)}{3} = \frac{(k-1) \cdot k \cdot (k+1)}{6}$, определяющая минимальное количество необходимых для расположения кораблей клеток в зависимости от k, монотонна. Значит, требуемое для решения задачи значение k можно найти двоичным поиском за $O(\log n)$.

Разбор задачи «Пасьянс»

Для начала осознаем, что максимальную последовательность Рик получит, если колода будет отсортирована в возрастающем порядке. Действительно, так как по условию требовалось, чтобы числа на карточках были в возрастающем порядке, любая последовательность подходящая под условие является подпоследовательностью отсортированной колоды.

Пусть после сортировки получится так, что все соседние карточки разной четности. В таком случае ответ будет равен размеру колоды. Теперь рассмотрим ситуацию, в которой две соседние карточки имеют одинаковую четность. Так как, как минимум одна из них не войдет в наибольшую последовательность, мы можем убрать из них и решить задачу для оставшейся колоды, так как для нее ответ будет такой же.

Реализация будет довольно простой и не будет требовать удаления из середины. Мы будем поддерживать массив, в котором будем хранить ответ, изначально положим туда минимальное число из колоды. Затем последовательно пройдемся по колоде, и если текущее число, отличается по четности от последнего взятого в ответ числа, возьмем его тоже. Остается только вывести размер получившегося ответа.

Суммарное время этого алгоритма $O(n \log n + n)$ или $O(n^2 + n)$, в зависимости от того, какую сортировку использует решение.

Разбор задачи «Капли»

Посчитаем, сколько раз каждая капля упадет между двумя последовательными очистками трубы и просуммируем эти значения по всем каплям: $sum = \sum_{i=1}^{n} \lfloor \frac{k}{p_i} \rfloor$. Также посчитаем, сколько капель упадет с каждой трубы за последние $t \bmod k$ секунд сбора капель и просуммируем эти значения: $rem = \sum_{i=1}^{n} \lfloor \frac{t \bmod k}{p_i} \rfloor$. Затем вычислим, сколько очисток трубы произойдет за t секунд: $cnt = \lfloor \frac{t}{k} \rfloor$.

Ответом является следующая сумма: $cnt \cdot sum + rem$.

Разбор задачи «Газорпазорп»

- Если k нечетное, то всегда выигрышной является стратегия «назвать предыдущее число». Действительно, тогда мы всегда будем называть его последними, и когда мы назовем число $k - 1$ раз, первому игроку придется назвать какое-то другое число, а мы продолжим нашу стратегию.

- Если k четное, то сначала предподсчитаем массив выигрышных-проигрышных позиций (тут нам наоборот называть когда-либо предыдущее число нет смысла, потому что по четности мы проиграем). Если хотя бы одна из позиций $x_{prev} - 1..x_{prev} - m$ — проигрышная, назовем это число и выиграем. Если нет, мы точно проиграем при оптимальной игре первого игрока и можно сразу же сказать «`I'm giving up`». Замечание: если первый игрок после нашего хода называет то же самое число, ответим ему тем же — теперь он в итоге проиграет по четности.

Разбор задачи «Морти и пароль»

Находим каждый элемент максимально возможной перестановки по очереди, начиная с самого первого. На его место может встать один из первых трех элементов. Берем из них максимальный, перемещаем его влево, если нужно. Считаем, сколько раз каждый из элементов уже был перемещен, и, опираясь на это, рассматриваем возможные варианты для элемента, который должен располагаться на следующей позиции и так далее. Заметим, что на каждом шаге будут израсходованы операции не более, чем у двух следующих элементов.

Разбор задачи «Матрица Рика»

Матрица, которая должна получиться, однозначно задаётся своей верхней левой подматрицей размера $\lceil \frac{n}{2} \rceil$ на $\lceil \frac{m}{2} \rceil$.

Переберём клетку из этой подматрицы. Рассмотрим те клетки, которые должны быть ей равны, то есть такие, которые получаются из неё отражениями относительно центральной горизонтали или вертикали. Таких клеток может быть четыре, две или одна. Возьмём значение, которое встречается среди них большее число раз, и присвоим его остальным.

Разбор задачи «Портальная пушка»

Переберем индекс i массива A, тем самым зафиксировав a_i. Для фиксированного i нам нужно посчитать $\sum_j (i-j) \cdot |a_i - b_j|$. Теперь раскроем $|a_i - b_j|$, рассмотрев два случая:

- $a_i \geqslant b_j$. Тогда $\sum_j (i-j) \cdot |a_i - b_j| = \sum_j (i \cdot a_i - i \cdot b_j - j \cdot a_i + j \cdot b_j)$;
- $a_i < b_j$. Тогда $\sum_j (i-j) \cdot |a_i - b_j| = \sum_j (i \cdot b_j - i \cdot a_i - j \cdot b_j + j \cdot a_i)$.

Выражения в обоих случаях одинаковы с точностью до знака, поэтому мы рассмотрим подсчет только первого из них. $\sum_j (i \cdot a_i - i \cdot b_j - j \cdot a_i + j \cdot b_j) = \sum_j i \cdot a_i - \sum_j i \cdot b_j - \sum_j j \cdot a_i + \sum_j j \cdot b_j$. Отсортируем массив пар (b_i, i) по возрастанию и двоичным поиском найдем минимальный индекс min_j, такой, что $a_i \geqslant b_{min_j}$.

- При фиксированном i, $i \cdot a_i$ — константа для всех j, поэтому $\sum_j i \cdot a_i = i \cdot a_i \cdot (|b| - min_j + 1)$;
- $\sum_j i \cdot b_j = i \cdot \sum_{j=min_j..|b|} b_j$;
- $\sum_j j \cdot a_i = a_i \cdot \sum_{j=min_j..|b|} b_j.index$, где $b_j.index$ — индекс b_j в начальном неотсортированном массиве (то есть второе число в паре);

Заметим, что чтобы посчитать сумму $\sum_{j=min_j..|b|} b_j$, достаточно посчитать префиксные суммы $pref_i = b_1 + b_2 + ... + b_i$ на отсортированном массиве. Чтобы посчитать суммы $\sum_{j=min_j..|b|} b_j.index$ и $\sum_{j=min_j..|b|} j \cdot b_j$, также достаточно посчитать другие префиксные суммы на отсортированном массиве. Таким образом, мы научились для фиксированного i за $O(\log |b|)$ вычислять сумму $\sum_j (i-j) \cdot |a_i - b_j|$. Суммарное время работы — $O(|a| \cdot \log(|b|))$.

Разбор задачи «Спасите Землю»

Рассмотрим кратчайшие отрезки от точки до одного круга и до другого. Если отрезок задевает оба круга, обновим ответ.

Теперь посмотрим, как выглядит решение. Сначала из точки нужно провести отрезок до какой-то точки на одной окружности, потом из этой точки провести отрезок до точки на второй окружности по направлению к центру этой окружности. Второй отрезок надо проводит в направлении центра второй окружности, потому что требуется из какой-то точки за минимальное расстояние дойти до окружности. Заметим, что ответом будет $|a-p| + |c-p| - r_c$, где a — начальная точка, p — точка на первой окружности, c — центр второй окружности, r_c — ее радиус. Следовательно, нужно минимизировать $|a-p| + |c-p|$, потому что r_c — константа.

Рассмотрим порядок, в котором будут посещены окружности. Найдем минимум требуемой функции. Это можно сделать с помощью тернарного поиска по углу точки p из b, где b — центр первой окружности, выбрав в качестве границ тернарного поиска углы точек a и c из b.