

# Арифметическое упражнение

Автор и разработчик задачи: Андрей Пархаев

## Подгруппа 1.

Пусть все  $x_i$  равны  $x$ . В первой подгруппе предлагается на каждом шаге применять изменение к нулевым элементам массива, если это возможно. Если же нулевых ячеек нет, будем применять изменения к самому последнему элементу массива. Тогда в последней ячейке будет чередоваться значение  $x, -x, x, \dots$

## Подгруппа 2.

Существует  $2^m$  различных способа сделать последовательность изменений, т.к. каждая из  $m$  операций применяется либо к первому, либо ко второму элементу  $a$ . Переберем всевозможные последовательности изменений и посчитаем сумму после их применения. Среди всех таких сумм возьмем максимальную. Такое решение работает за  $O(m \cdot 2^m)$ .

## Подгруппа 3.

Воспользуемся динамическим программированием. Пусть  $dp[pref][i][j] = true/false$  — можно ли обработать первые  $pref$  изменений, чтобы выполнялось  $a_1 = i, a_2 = j$ . Обратите внимание, что  $i$  и  $j$  могут быть отрицательными. Понятно, что  $a_1$  и  $a_2$  не могут по модулю превысить сумму значений  $x_i$ . Сумма всех  $x_i$  не превосходит  $10m$ , поэтому мы можем хранить значения в диапазоне от  $-500$  до  $500$ .

## Подгруппа 4.

Улучшим динамику из предыдущей группы. Заметим, что при фиксированном значении  $a_1$  на префиксе, нам имеет смысл пересчитываться только через минимальный или максимальный доступный  $a_2$ . Поэтому будем хранить явное значение только одного из элементов массива. Заведем  $dp_{min}[pref][i]$  и  $dp_{max}[pref][i]$ , которые хранят минимальное и максимальное достижимое значение  $a_2$  на префиксе, если  $a_1 = i$ .

## Замечание.

Сделаем замечание, которое поможет нам существенно продвинуться к решению. Понятно, что каждый элемент последовательности  $x_i$  войдет в итоговую сумму либо со знаком «+», либо со знаком «-». Давайте разобьем все  $m$  элементов на  $n$  последовательностей, в зависимости от того, к какой позиции массива  $a$  было применено изменение. Некоторые из последовательностей могут быть пустыми. В каждой последовательности знаки элементов чередуются, и последний элемент всегда имеет знак «+». А потому, если заменить знаки на  $+1$  и  $-1$  соответственно, то сумма на каждом суффиксе таких массивов будет равна либо 1, либо 0. Совместим все последовательности обратно в одну и посмотрим на сумму знаков на произвольном суффиксе. Она будет находится в диапазоне  $[0; n]$ . Теперь мы можем использовать это, как критерий для проверки последовательности знаков на правильность.

## Подгруппа 5.

Переберем знак для каждого  $x_i$  и проверим, что на каждом суффиксе выполняется описанный критерий. Решение работает за  $O(m \cdot 2^m)$ .

## Подгруппа 6 и 7.

Снова воспользуемся динамикой. Пусть  $dp[i][j]$  — максимальная сумма, которую можно получить на суффиксе  $i$ , если текущий баланс знаков равен  $j$ . Тогда  $dp[i][j] = \max(dp[i+1][j-1] + x_i, dp[i+1][j+1] - x_i)$ . Такое решение работает за  $O(nm)$ .

## Подгруппа 8.

В этой подгруппе предлагается действовать жадно, соблюдая критерий баланса. Например, можно идти с конца последовательности  $x_i$ . На каждом шаге будем пытаться взять новый элемент и, при надобности, выкидывать какой-то элемент, который был взят ранее. Т.к. различных значений не более двух, выкидывать имеет смысл только минимум. Будем поддерживать в очереди минимумы, которые были взяты со знаком «+» и которые еще можно выкинуть.

## Подгруппа 9 и 10.

Существует несколько альтернативных подходов. Рассмотрим несколько из них.

1. Улучшим решение для предыдущей подгруппы. Будем идти с конца, храня текущую расстановку знаков для элементов суффикса. В дереве отрезков будем поддерживать элементы, которым

мы можем изменить знак на противоположный. Когда переходим к новому элементу, пытаемся поставить ему знак «+» (в случае, если это невозможно, меняем знак минимального элемента со знаком «+», для которого это можно сделать) и смотрим, как меняется сумма от этого действия. Аналогично, пробуем поставить знак «-» и смотрим, как меняется сумма. Среди двух вариантов, выбираем тот, в котором сумма будет наибольшей.

2. Отсортируем последовательность  $x_i$  по убыванию математического модуля. Будем идти по отсортированным значениям и на каждом шаге пытаться поставить элементу нужный знак (если число положительное — «+», иначе «-»), если это возможно. Иначе будем ставить тот, знак, который мы вынуждены поставить. Чтобы понимать, можем ли мы поставить тот или иной знак, будем хранить дерево отрезков, в каждой позиции которого хранится текущий баланс соответствующего суффикса. Если мы хотим поставить очередной знак, надо посмотреть на минимальный и максимальный баланс который уже достигается перед ним и проверить, можно ли расставить оставшиеся знаки так, чтобы критерий баланса не нарушался.

3. Вспомним решение для подгрупп 6 и 7, использующее динамическое программирование. Заметим, что функция выпуклая отдельно по двум четностям, поэтому мы можем воспользоваться slope trick.