Задача A. Наибольшее число

Решать задачу будем так: пройдёмся слева направо по нашей строке s, поддерживая сколько цифр мы ещё можем изменить. Рассмотрим i-ую цифру: если она меньше, чем своё противоположное - меняем. Код: https://pastebin.com/2ZzAJSVa.

Задача B. Сумма произведений

Посмотрим например какой вклад в ответ даёт a_1:

$$a_1 * a_2 + a_1 * a_3 + ... + a_1 * a_n$$

Можно вынести a_1 за скобки :

$$a_1 * (a_2 + a_3 + ... + a_n)$$

Получается, достаточно найти сумму всех чисел, и найти вклад каждого числа в итоговый ответ становится уже куда проще: когда будем рассматривать a_i, от суммы всех чисел отнимем это число, и к ответу добавим a_i * sum (где sum - сумма всех чисел массив, кроме a_i).

Код https://pastebin.com/bbk1Nx8K

Задача C. Как можно больше!

Воспользуемся динамическим программированием для решения задачи.

Очевидно, круглость числа — это минимум из степеней 2 и 5 в числе. Пусть $pw5_i$ — это максимальная степень 5 в числе, $pw2_i$ — максимальная степень 2.

Пусть $dp_{i,j,l}$ — это максимальное количество двоек, которые мы можем собрать, проверив i чисел, взяв j из них с суммарной степенью пяти равной l. Обычно такое называют «Задача о рюкзаке».

Есть два типа переходов. Можно либо взять текущий элемент, либо пропустить его:

- $dp_{i+1,j+1,l+pw5_i} = \max(dp_{i+1,j+1,l+pw5_i}, dp_{i,j,l} + pw2_i)$
- $dp_{i+1,j,l} = \max(dp_{i+1,j,l}, dp_{i,j,l})$

Ответ будет максимумом из $\min(i, dp_{n,k,i})$ по всем i. Поддержка столь большого количества состояний может привести к ML, надо держать первое измерение в два слоя и пересчитывать на лету.

Код https://pastebin.com/G4kpstTH

Задача D. Капи-капи-капибара

Прежде чем решать эту задачу, необходимо понимать что такое Дерево Фенвика, а так же сжатие координат.

Что такое сжатие координат? У нас есть массив a их n чисел, где каждое $a_i \leqslant 10^9$. Мы хотим получить такой массив c, что выполнятся несколько условий:

- Если для каких-то i и j $(i < j)$ выполнялось: $a_i < a_j$, то выполняется и $c_i < c_j$
- Если для каких-то i и j $(i < j)$ выполнялось: $a_i > a_j$, то выполняется и $c_i > c_j$
- Если для каких-то i и j выполнялось: $a_i = a_j$, то выполняется и $c_i = c_j$
- Все $c_i \leqslant n$

Как всё же сжимать координаты? Давайте отсортируем массив a и удалим в нём одинаковые числа. Теперь когда мы хотим узнать сжатую версию для какого-то a_i, мы просто бинарным поиском ищем этот элемент в отсортированном массиве a. Позиция этого числа и будет его сжатой версией.

Теперь приступим к решению самой задачи. Мы сожмём сразу все точки по x и по y. Теперь по оси OX построим дерево фенвика. Если в обычном дереве фенвика у нас в ячейке i лежала сумма чисел на отрезке $[f(i), i]$, то теперь мы будем хранить в i-ой ячейке дерева фенвика ещё одно дерево фенвика, в котором лежат все точки, чьи сжатые координаты x лежат на отрезке $[f(i), i]$. Теперь как обрабатывать запросы? Легко. Мы прыгаем по координатам x как в обычном дереве фенвике, и в каждом дереве фенвика ищем сумму на префиксе от 0 до y. Каждая точка входит в $logN$ деревьев

фенвика, и итоговая асимптотика на запрос суммы будет log^2N Аналогично и с обновлением в точке. Разница от двумерного дерева фенвика в том, что в этом случае мы занимаем лишь $NlogN$ памяти, вместо N^2.

Код: https://pastebin.com/4wsjAwGw