Задача A. Рик и Морти, а также злой таможенник

Заметим, что если $n = m = 1$, то ответ существует, так как у нас просто нет двух пар строк и столбцов, условие таможенника нарушиться не может, ответ в этом случае - YES.

В случае $n = 1, m \neq 1$, то таблица, которую нам необходимо заполнить, является строкой длины m. Тогда, в этой строке возьмём столбцы, в которых записаны числа 1 и 2 (в нашей таблице есть все числа от 1 по $n \cdot m = m \geqslant 2$ включительно. Значит в ней обязательно есть числа 1 и 2). Произведения чисел в этих столбцах равно самим числам (потому что каждый столбец имеет высоту $n = 1$), значит произведения чисел в этих 2 столбцах - не взаимно простые числа, противоречие. Значит, в таком случае ответ NO.

Случай $n \neq 1, m = 1$ аналогичен случаю $n = 1, m \neq 1$ (будем рассматривать не столбцы, в которых находятся числа 1 и 2, а строки), ответ - NO.

Осталось рассмотреть случай $n \neq 1, m \neq 1$. Отсюда следует, что $n \geqslant 2$ и $m \geqslant 2$. Давайте в таком случае поставим на клетки $a_{1,2}, a_{1,3}, \cdots, a_{1,m}$ числа $2, 4, 6, \cdots, 2 \cdot (m-1)$. А в клетки $a_{2,1}, a_{3,1}, \cdots, a_{n,1}$ поставим числа $2 \cdot m, 2 \cdot (m+1), \cdots, 2 \cdot (m+n-2)$. Сначала докажем, что мы так можем сделать. Так как все числа различны и положительны, нам необходимо только доказать, что последнее число не превосходит $n \cdot m$ - отсюда будет следовать, что все подобранные числа лежат в диапазоне чисел, который просит таможенник. То есть надо доказать, что $2 \cdot m + 2 \cdot n - 4 \leqslant n \cdot m \Leftrightarrow 2 \cdot m - 4 \leqslant n \cdot (m-2) \Leftrightarrow 0 \leqslant (n-2) \cdot (m-2)$ - очевидно, так как $n \geqslant 2$ и $m \geqslant 2$. Теперь покажем, что если мы расставим все оставшиеся числа в любом порядке, то условие охранника всегда будет верно:

- Рассмотрим произведения любых 2 строк. Каждое из них будет чётным, так как в каждой строке есть хотя бы 1 чётное число (мы поставили чётные числа $2, 4, \cdots, 2 \cdot (n+m-2)$ таким образом). А значит, эти 2 числа не будут взаимно простыми - они оба делятся на 2
- Рассмотрим произведения любых 2 столбцов. Аналогично прошлому пункту получаем, что в каждом из них произведение будет чётным $\Rightarrow$ 2 таких числа не взаимно просты

Мы привели пример таблицы, которая удовлетворяет требованиям таможенника, а значит, ответ - YES.

Задача B. Нейтрино бомба

По условию задачи элементы a_i и a_{i+1} можно менять местами, если их сумма нечётна, то есть если одно из них чётное, а другое - нечётное.

Заметим, что если $i < j$, и при этом a_i и a_j одинаковой чётности, то элемент a_i никогда не сможет стать правее a_j. Поэтому выпишем по порядку в вектор *odd* все нечётные элементы массива a, а в вектор *even* все чётные. Если какой-то из векторов *odd* и *even* оказался не отсортированным, то отсортировать массив a невозможно.

Если вектора *odd* и *even* оказались отсортированными, то отсортировать массив a можно, например, сортировкой пузырьком. При этом все действия будут корректны, т. к. нам никогда не потребуется менять местами два элемента одинаковой чётности.

Задача C. Интересные числа

Назовем число *интересным*, если для любой пары соседних цифр ни одна из них не делится на другую. Каждую секунду, если число на доске не *интересное*, то его длина уменьшается на 1, иначе оно остается неизменным. Следовательно, через N секунд на доске останутся только *интересные* числа из отрезка от 0 до 10^N. При этом все *интересные* числа от 0 до 10^N изначально были написаны на доске, следовательно, они будут и в конце. Число 10^N не интересное, значит оно не останется в конце. То есть надо найти количество *интересных* чисел, в десятичной записи которых не более N цифр.

Посчитаем это значение с помощью динамического программирования (ДП). Будем вести ДП по цифрам. $dp[i][j]$ – количество i-значных *интересных* чисел, которые заканчиваются на цифру j. Для $i = 1$ все значения равны 1. Пересчет динамики: перебираем все $1 < i < N$, для каждого j перебираем $0 \leqslant j < 10$. Чтобы посчитать $dp[i][j]$, перебираем следующую цифру (цифра k). Если для

j и k верно, что ни одно из них не делится на другое, то прибавляем к $dp[i][j]$ значение $dp[i-1][k]$, то есть количество чисел, заканчивающихся на jk.

После подсчета динамики остается сложить все значения. Данный алгоритм работает за $O(A^2 \cdot n)$, где $A = 10$. (То есть алгоритм работает за $O(n)$, но надо учитывать размер константы.)

Задача D. Сверхреалистичная игра

Длина строки s хотя бы 4 и всегда чётная. Поэтому рассмотрим 4 символа $ABCD$, которые стоят ровно посередине строки. Рассмотрим 3 случая:

- Если $B = C$, то выиграл Морти, потому что он может следовать такой стратегии: каждый раз, когда Рик убирает букву слева, Морти будет убирать букву справа, и наоборот.

  Таким образом, если игра дошла до последних двух символов, то это символы BC и Морти выиграет, т. к. они равны.

- Если $A = B$ и $C = D$, то тоже выиграет Морти, потому что он может следовать такой стратегии:

  Своим первым ходом он повторит ход Рика: т. е. если Рик убрал букву слева, то Морти тоже уберёт букву слева, а если Рик убрал справа, то Морти тоже. А каждым своим следующим ходом Морти будет действовать по стратегии из предыдущего пункта, т. е. каждый раз когда Рик убирает букву слева, Морти будет убирать букву справа, и наоборот.

  Таким образом, если игра дошла до последних двух символов, то это либо символы AB, либо символы CD и Морти выиграет, т. к. они равны.

- Если $B \neq C$ и либо $A \neq B$, либо $C \neq D$, то выиграет Рик. Пусть, не умаляя общности, $A \neq B \neq C$.

  Тогда первым своим ходом Рик уберёт последнюю букву. А каждым своим последующим ходом, когда Морти убирает букву слева, Рик будет убирать букву справа, и наоборот.

  Таким образом, после каждого хода Морти посередине строки будут находиться либо символы AB, либо символы BC. Поскольку эти символы различные, строка никогда не окажется палиндромом, и Рик выиграет.

В решении достаточно рассмотреть эти 3 случая и вывести соответствующее имя.

Задача E. Мистер Мисикс

Заметим, что если есть лунка с координатами (x_1, y_1) и лунка с координатами (x_2, y_2), при этом $x_1 \leqslant x_2$ и $y_1 \leqslant y_2$, то про лунку (x_2, y_2) можно забыть, т. к. любой шар, который попадает в неё, сможет попасть и в лунку (x_1, y_1).

Поэтому давайте сначала забудем про все лунки, с описанным выше условием. Это можно сделать при помощи следующего алгоритма: положим все лунки в вектор a, отсортируем его по координате x по возрастанию, а при совпадении - по координате y по возрастанию. Создадим второй вектор b, в который будем класть выбранные лунки. Переберём все лунки (x_i, y_i) из вектора a и будем класть их в вектор b, если он пустой, или если координата y последней точки, положенной в b, больше, чем y_i.

Заметим, что в результате в векторе b будут точки, у которых все координаты x идут строго по возрастанию, координаты y идут строго по убыванию, при этом мы выкинули только ненужные лунки.

Теперь, нам для каждой предлагаемой точки (a_i, b_i) нужно проверить, есть ли в векторе b такая точка (x_j, y_j), что $x_j \leqslant a_i$ и $y_j \leqslant b_i$. Для этого при помощи бинарного поиска найдём точку (x_j, y_j), для которой $x_j \leqslant a_i$ и j максимально. Если для этой точки выполнено $y_j \leqslant b_i$, то ответ YES. Если для этой точки $y_j > b_i$, то никакая лунка до j-й не подходит, т. к. у них больше координата y, а никакая лунка после j-й не подходит, т. к. у них координата x больше a_i, поэтому ответ NO.

Итоговая асимптотика $O((n+q)log\,n)$

Задача F. Капсулы времени

Для начала проверим, есть ли у нас год, который не покрывается ни одной капсулой. Если такого года нет, то ответ равен -1. В противном случае давайте рассмотрим самый левый и самый правый года, не покрытые капсулами. Назовем такие года пропусками.

Очевидно, что нам нет смысла изменять капсулы, которые находятся между этими пропусками. Без ограничения общности давайте найдем все подходящие капсулы (то есть те капсулы, которые можно изменить так, чтобы все года были покрыты) слева от первого пропуска. Во-первых, их длина должна позволять покрыть пропуски. Во-вторых, нам нужно понять, насколько далеко мы можем сдвинуть эту капсулу, то есть нужно найти первый год, больший левой границы капсулы, такой, что он покрыт только этой капсулой. Если такого года нет, то мы можем свободно перемещать эту капсулу, иначе мы не можем сдвинуть её правее этого года, так как тогда этот год будет не покрытым. Зная, насколько сильно вправо можно сдвинуть капсулу, мы можем проверить для неё, будет ли она покрывать все пропуски.

Аналогично можно проверить все отрезки, находящиеся справа от последнего пропуска. Всё это можно реализовать за асимптотику $O(n \log n)$, используя технику сканирующей прямой. Естественно, среди всех подходящих капсул нас интересует капсула с минимальной стоимостью.

Задача G. Межпространственное дерево

Пусть мы как-то расставили a_i. Тогда давайте поймем, как посчитать ответ. Если мы будем знать, сколько путей проходит по каждому ребру, то ответ будет в точности сумма произведений $b_i \cdot cnt_i$, где b_i — число, которое распределили i-му ребру, cnt_i — сколько путей по нему проходит.

Тогда это наталкивает на решение, что мы хотим отдать большие a_i рёбрам с маленькими cnt_j и наоборот. Тогда решение заключается в том, что считаем для каждого ребра, сколько путей по нему проходит. После этого сортируем рёбра в порядке возрастания количества путей, а a_i сортируем в порядке убывания, а после этого считаем ответ $a_i \cdot cnt_i$.

Чтобы посчитать количество путей, проходящих через ребро, можно воспользоваться обходом в глубину. Тогда мы сможем посчитать для каждой вершинки размер её поддерева. Тогда количество путей, проходящих через ребро, это будет произведение количества вершин, находящихся по разные стороны этого ребра. Тогда, если мы знаем размер поддерева v, то мы сможем посчитать количество путей, проходящих через ребро, в которое мы попали в эту вершину, как $size_v \cdot (n - size_v)$, где $size_v$ — размер поддерева v.

Задача H. Чифир

Поймем как меняется уровень жидкости у грани, в сторону которой мы наклоняем, и у противоположной ей. Пусть у первой уровень увеличился на $h_1 \leqslant (c - h)$, а у второй уменьшился на $h_2 \leqslant h$. Тогда рассмотрим часть стакана на высоте от $h - h_2$ до $h + h_1$. Заметим, что это прямоугольный параллелепипед, и уровень жидкости делит его на 2 равные части, то есть объем жидкости равен половине объема рассматриваемого параллелепипеда, то есть $ab(h_1+h_2)/2$. При этом изначально объем жидкости в этой части стакана был такой же и был равен abh_2. Следовательно $ab(h_1+h_2)/2 = abh_2$, то есть $h_1 + h_2 = 2h_2$. Значит $h_1 = h_2$.

Тогда если $h \leqslant c$, то при $h_1 = h_2 = h$ мы сможем увидеть отметку 0 – минимальную из возможных отметок. А иначе наибольшее значение для h_1 (а значит и для h_2) равно $c - h$. Значит наименьшая отметка, которую можно увидеть, это $h - (c - h) = 2h - c$.

Задача I. Забор Джина

Очевидно, что если мы можем получить m досок одинаковой длины, то $m - 1$ доску одинаковой длины мы тоже можем получить. Поэтому сделаем бинарный поиск по ответу.

Пусть внутри бинпоиска мы хотим проверить, можно ли получить m досок одинаковой длины. Пусть мы получили некоторые m досок длиной L. Тогда одна из этих досок изначально имела длину L. Потому что если это не так, то каждая из m досок была удлинена/укорочена на ненулевую длину. Пусть, не умаляя общности, u досок были удлинены, v досок укорочены и $u \leqslant v$. Тогда выберем L' - минимальную изначальную длину из тех досок, которые были укорочены. Тогда сделать весь этот набор из m досок длиной L' будет стоить не дороже, чем сделать длиной L, при этом одна доска не изменит свою длину.

Также отсортируем массив a по неубыванию. Тогда очевидно, что если мы можем получить какие-то m досок одинаковой длины, то мы можем рассматривать только доски, идущие подряд в массиве сортировки.

Таким образом, нам нужно выбрать некоторые индексы $1 \leqslant l \leqslant i \leqslant r \leqslant n$, такие, что ответом будут все доски с l по r, то есть $r - l + 1 = m$, и в итоге их длина будет равна $a[i]$. Если мы предподсчитаем массив префиксных сумм $pref$, то стоимость будет равна $cost(l, r, i) = a[i] * (l - i + 1) - (pref[i] - pref[l - 1]) + (pref[r] - pref[i]) - a[i] * (r - i)$.

Будем перебирать все возможные l слева-направо, при этом находя такое i, что для него стоимость будет оптимальна. Тогда очевидно, что i тоже будет двигаться только вправо, и внутри каких-то l и r его можно жадно двигать вправо, пока стоимость улучшается. Поэтому после перемещения l вправо на единицу будем двигать i вправо, пока стоимость становится лучше. Проверять, становится ли стоимость лучше, можно за $O(1)$ по указанной выше формуле $ans(l, r, i)$. Т. к. индексы l, r, i двигаются только вправо, то асимптотика такого перебора $O(n)$.

Из всех возможных позиций l, r найдём ту, у которой стоимость лучше, и проверим, превосходит ли она k в бинпоиске.

Итоговая асимптотика $O(n \log n)$.

Задача J. Космический светофор

Ограничения на n в этой задаче до 4, поэтому возможных вариантов ответа не более 10^4. Переберем итоговый ответ и нужно научиться проверять, могло ли быть такое исходное число. Нужно перебрать все времена просыпания и проверить, что текущее время на светофоре отображается также. Такую проверку можно сделать за $O(7m)$ действий. Итоговая сложность $O(7 \cdot 10^n \cdot m)$. Также с помощью битовых операций можно убрать цикл по индикаторам и получить время работы $O(n \cdot m)$