Задача A. Три программиста

Заметим простой факт: если в каждые три последовательных дня должны дежурить три различных программиста, то они все будут должны дежурить по очереди, которая определяется их распределением в расписании на первые три дня. Всего существует 6 распределений на первые три дня, чтобы каждый программист был на смене по одному разу, значит достаточно для каждого расписания на первые три дня построить расписание целиком, посчитать суммарное количество решённых задач и найти максимальную сумму для всех распределений, а также вывести его.

Задача C. Министерство магии

Обозначим длину номера как строки L. Если $L = 1$, то нам не подходит не один из кабинетов, потому что номер - это конкатенация двух ненулевых строк. Если $L > 1$, то существует ровно $L - 1$ способ поделить эту строку на две - номер этажа и номер кабинета на этом этаже, однако нужно проверить, что оба эти номера не имеют ведущих нулей. Если номер этажа по формату ввода не может иметь лидирующих нулей, то номер кабинета на этаже из строки может обладать таким свойством, тогда нужно просто не считать этот вариант в ответ. Поскольку мы перебираем всевозможные разделения строки на две, то вторая строка может начинаться с любого символа за исключением первого, если она начинается не с нуля, то такой кабинет существует. Пусть нулей в строке M штук (причём они не могут быть лидирующими по формату ввода), тогда ответ будет равен $L - 1 - M$.

Задача D. Амбал и пакетик

Сэмулируем процесс. Для этого явно заведем переменную W — используемый объем пакетика. Обновление этой переменной будем производить, как в условиях задачи.

Остается только ответить, что оптимальнее делать, когда приходит амбал: ужимать очередную вещь или содержимое пакетика. На i-й операции, когда $v_i + W \geqslant V$, отметим, что необходимо уменьшить как можно сильнее за одну помощь амбала суммарный объем $v_i + W$. Уменьшение делением на 2 тем больше, чем больше делимое. Поэтому, если $v_i > W$, нужно уменьшать вещь, а иначе — содержимое пакетика (при равенстве можно делать любую операцию).

Задача E. Кроссворд

Нам нужно расставить буквы так, чтобы гласные и согласные буквы чередовались. Заметим, что нам не надо выбирать сами буквы — достаточно посчитать количество гласных и согласных. Но если мы зафиксируем гласность или согласность одной буквы в слове, то для всех остальных букв в слове гласность или согласность определяется автоматически.

Также, поскольку кроссворд связен, по гласности или согласности одной буквы можно определить все остальные буквы в кроссворде. Получается, есть всего два варианта расставить буквы.

Есть простой способ определить гласность букв в расстановке. Раскрасим всё поле в шахматном порядке. То есть, клетку (r_i, c_i) покрасим в чёрный или белый цвет в зависимости от значения $r_i + c_i \bmod 2$. Тогда либо на чёрных клетках будут все гласные буквы, на белых — все согласные; либо наоборот.

Это позволяет нам посчитать ответ. Пусть в кроссворде всего A чёрных клеток и B белых. Тогда в одном случае будет A гласных букв и B согласных. Отсюда количество способов расставить сами буквы равно $5^A \cdot 20^B$, так как всего используется 5 различных гласных и 20 различных согласных букв. В другом случае, аналогично, будет $20^A \cdot 5^B$ способов расставить буквы.

Получается, ответ на задачу равен

$$5^A \cdot 20^B + 20^A \cdot 5^B.$$

Это можно ещё больше упростить. Заметим, что $5 \cdot 20 = 100$. Тогда, если $min(A, B)$ — наименьшее из чисел A и B, $max(A, B)$ — наибольшее из них, то это же количество способов равно

$$100^{min(A,B)} \cdot (5^{(max(A,B)-min(A,B))} + 20^{(max(A,B)-min(A,B))}).$$

Это можно посчитать даже не прибегая к длинной арифметике: $5^{(max(A,B)-min(A,B))} + 20^{(max(A,B)-min(A,B))}$ вычислим явно, а затем припишем в конец $2 \cdot min(A, B)$ нулей.

Задача F. В тренде

В этой задаче достаточно сравнивать оценки в виде пар целых чисел, где первый элемент означает оценку, а второй - 1 при модификаторе «`light`», 2 при модификаторе «`decent`», 3 при модификаторе «`strong`». Тогда приоритетным сравнением будет сравнение первых элементов (то есть самой оценки из условия задачи), а при равных оценках будут сравниваться числа, соответствующие модификаторам, что и является сравнением из условия задачи. Осталось определить, какая из оценок будет находиться между другими. Это можно сделать как простыми условными операторами, так и сортировкой всех оценок и последующим выбором второго элемента.

Задача H. Лучший мой подарочек

Отсортируем ценности $a_1, a_2, \ldots, a_n$ по убыванию, и получим новый массив $b_1, b_2, \ldots, b_n$. Отметим, что сортировку нужно произвести одним из быстрых способов, например, при помощи встроенной сортировки или при помощи сортировки подсчетом. Квадратичные сортировки, такие как пузырьковая, не подойдут из-за высоких ограничений на n.

Пусть ребятам никогда не попадаются свои же подарки. Тогда заметим, что ребенок с номером i должен получить i-й по ценности подарок, то есть подарок с ценностью b_i. Значит, числа $b_1, b_2, \ldots, b_n$ были бы ответом, если бы ценность своего же подарка не равнялась нулю.

Все ценности подарков различны, а значит подарок с ценностью a_i единственный. Значит, если ребенок должен взять подарок $b_i = a_i$, то это и есть его подарок. Но тогда для него ценность этого подарка равна нулю, и он возьмет максимальный по ценности, но не этот, а значит он возьмет подарок с ценностью b_{i+1}. Единственное исключение: если $i = n$ — в таком случае ребенок все равно возьмет подарок с ценностью b_i.

Пусть ребенок с номером i взял подарок с ценностью b_{i+1}. Заметим, что $(i+1)$-му ребенку подарок с номером i гарантированно не принадлежит (так как он принадлежит i-му ребенку), а значит он точно возьмет его. Получается, что если $b_i = a_i$, то просто i-й и $(i+1)$-й ребенок меняются подарками.

Таким образом, после сортировки реализация сводится к следующему: переберем номера ребят, и для каждого индекса i, что $a_i = b_i$, если i — не последний номер, поменяем местами b_i и b_{i+1}. Получившийся массив и будет ответом на задачу.

Задача I. Интервью

Переформулируем задачу: нужно посчитать количество уникальных элементов одного массива. Это можно сделать с помощью сортировки или сета за $O(N \log N)$. Это можно улучшить. Максимальное значение пикантности равняется 10^5, что позволяет использовать сортировку подсчётом за $O(N)$.

Задача J. Эпидемия

Решим данную задачу следующим образом:

Построим неориентированный граф, где вершины — наши города, а рёбра — дороги. Упорядочим вершины в порядке возрастания t_i, для текущей рассматриваемой вершины проверяем, подсчитан ли для неё ответ, и в случае если нет — запускаем поиск в ширину от неё.

Однако есть некоторые нюансы, о которых стоит важно помнить. Во-первых, вершин подходящих такому условию может быть несколько, и важно не забыть их также добавить в очередь (получим запуск BFS с несколькими источниками). Из-за этого могут возникнуть трудности в том, чтобы безошибочно и упорядочить вершины, и не потерять какую-нибудь из них. Как вариант решения данной проблемы — складывать номера вершин, как список, в словарь по ключу времени. Но ограничения позволяли сделать тоже самое, но в форме массива.

Во-вторых, по условию задачи мы можем наткнуться на ситуацию, что вершины, нами рассматриваемые, достигли такой величины ответа, что она равна значению t_i очередного набора городов, что ещё не были посещены — их также нужно не забыть добавить в обход именно в этот момент, иначе мы либо испортим ответ, либо ещё хуже — замедлим решение задачи, т.к. придётся делать лишние проходы по графу и исправлять ответы раз за разом.

Таким образом, мы получаем решение в виде суммарно одного поиска в ширину, что даёт сложность решения $O(n + m)$.

Но помимо данного решения, есть альтернативное — создать фиктивные вершины, соответствующие «дням» из данных t_i, провести односторонние ребра вершины с равным ей значением времени, а также в $t_i + 1$. Такую возможность нам дают ограничения на t_i. Тогда, если запустить поиск в ширину, от вершины первого дня и подсчитать расстояние, это будет ответом для каждого города из исходных данных.