Задача A. Комплектация команд

Вначале соберем некоторое число полных команд из трех или двух человек — их число будет $[\frac{a}{3}] + [\frac{b}{2}]$. Теперь нам нужно использовать некоторых студентов из третьей группы, которым все равно на размер команды — в первую очередь добьем до полного состава команды, в которых осталось одно свободное место (например, если $a \mod 3 \equiv 1$ или $b \mod 2 \equiv 1$), затем распределяем оставшихся студентов из третьей группы в команды по два человека, так как тогда команд будет максимально много.

Сложность решения: $O(1)$.

Задача B. Точки и дуги

Пусть A_{2i} — количество способов соединить $2i$ точек дугами так, что дуги проведены только сверху. Это число равно $(i-1)$-му числу Каталана, что можно вычислить за константу или посчитать динамикой.

Пусть B_{2i} — количество способов соединить $2i$ точек дугами так, что дуги проведены как сверху, так и снизу. Переберем, сколько есть в паросочетании на $2i$ точках точек, которые соединены дугами сверху — их может быть 0, 2, 4, ..., $2i$ — пусть $C_{i,j}$ — количество способов выбрать из $2i$ точек $2j$ различных — это количество сочетаний из $2i$ по $2j$. Тогда $B_{2i} = \sum C_{i,j} \times A_{2j} \times A_{2i-2j}$ по всем $0 \leqslant j \leqslant i$.

Сложность решения: $O(n)$.

Задача C. Выпей отвар из трав

Будем использовать структуру данных «словарь» или «map», создадим словарь f и пройдемся по обоим мультимножествам частиц в котле a и котле b, считая каждый элемент из котла a со знаком «плюс», а из котла b со знаком «минус» — получили словарь f, в котором для каждого ключа известно, насколько больше таких элементов в котле a, чем в котле b.

Посчитаем сумму $s = \sum |f_i|$ по ключам f_i. Эта сумма показывает, сколько минимум нужно сделать операций, чтобы сравнять наборы ингредиентов в обоих котлах. Заметим, что так же котлы можно сравнять за $s+2k$ операций для некоторого натурального k, но никак не за $s+1+2k$. Поэтому после проверки на то, что $s \leqslant k$, должна следовать проверка на то, что $s \mod 2 = k \mod 2$ — только в таком случае мы можем сделать зелья в котлах одинаковыми.

Сложность решения: $O(n+m)$.

Задача D. Проверка стажера

Если $n = m$, то сразу выведем 0. Иначе будем рассматривать текущее число файлов в двоичном представлении.

В таком случае, при условии, что нельзя выполнять операции прибавления более одного раза подряд, применение операции `duplicate` равносильно битовому сдвигу влево.

Рассмотрим, какая операция могла быть первой:

- Если это операция `create`, то начинаем работать с числом $m+1$;
- Если это операция `duplicate`, то начинаем работать с числом $2m$.

В обоих случаях мы уже сделали одну операцию. Можно показать, что далее мы можем из начального числа $m+1$ или $2m$ получить n единственным образом и только тогда, когда двоичная запись начального числа совпадает с префиксом числа n в двоичной записи. Потому что теперь наши операции трансформировались следующим образом: мы можем дописать к числу «0» справа за одну операцию (умножая число на 2, т.е. делая битовый сдвиг влево на 1 разряд) или «1» за две операции (сначала умножая на 2, затем прибавляя 1).

Сложность решения: $O(log(n))$.

Задача E. Сидит ворона

Определим, когда ворона не видит столб i из-за столба j — только тогда, когда столб j ближе к вороне, чем столб i, и столбы i и j одновременно или слева, или справа от столба k, и когда угол,

образованный отрезком между столбом k и столбом j и самим столбом k, больше угла, образованного отрезком между столбом k и столбом i и самим столбом k. Находить и сравнивать сами углы нет необходимости — можно сравнивать их тангенсы или котангенсы как дроби с целочисленными числителями и знаменателями (обрабатывая знаки «плюс» и «минус» для тангенса).

Таким образом, можно сделать проход по столбам слева от столба k, начиная с самого столба k, а затем справа от столба k, начиная со столба $k+1$. При обоих обходах сохраняя наибольший угол и вычисляя, какие столбы остаются видимыми, а какие нет.

Сложность решения: $O(n)$.

Задача F. Минус три!

Рассмотрим самого правого матроса, который остался в ряду после применения k операций. Пусть его номер равен i. Для него верно, что были удалены $(n-i)$ матросов правее него (с большими номерами) и $(k-n+i)$ матросов левее него. При этом должно выполняться условие — матрос i имеет максимальный рост среди матросов с номерами i, $i+1$, $i+2$, ..., n. Если же были удалены все матросы с номерами меньше i, то также i будет иметь максимальный рост среди матросов с номерами $1...i-1$.

Ключевая идея решения заключается в применении динамического программировании. Опишем кратко основные моменты. Пусть $dp_{i,j}$ — минимальная сумма ростов матросов с 1 по i, что среди них было удалено ровно j матросов, при этом матрос i не был удален. Считаем, что соблюдается $0 \leqslant j < i \leqslant n$.

Тогда есть несколько баз:

$dp_{i,i-1} = a_i$ в случае, если $a_i = \max(a_1, a_2, ..., a_i)$, иначе $dp_{i,j-1} = \inf$.

$dp_{i,0} = a_1 + a_2 + ... + a_i$.

Переходы:

$dp_{i,j} = \min(a_i + dp_{i-1,j}, a_i + dp_{i-d,j-d+1})$ при всех таких $d \leqslant j$, что a_{i-d} или a_i максимум на отрезке от $i-d$ до i (так как мы удаляем всех матросов между $i-d$ и i), иначе просто $dp_{i,j} = a_i + dp_{i-1,j}$.

Ответом является такое минимальное $dp_{i,j}$, что a_i максимум на суффиксе от i до n, и $j + (n-i) = k$.

Сложность решения: $O(n^2 k)$.

Задача G. Наконец-то хэширование

Нужно проверить, сколько операций понадобится для каждого случая и вывести минимальное число. Пусть в введенном пароле есть x символов звездочки (в пароле не введены x цифр). Тогда есть следующие варианты решения задачи:

- $(n+x)$ — добиваем пароль до длины n, при этом пароль сбрасывается, и мы вводим его заново за n нажатий;
- $(2n-x)$ — стираем $n-x$ символов введенного пароля и вводим пароль заново за n нажатий;
- $(2n-2y-x)$ — где y максимальная длина общего префикса паролей, стираем $n-x-y$ цифр, остается введенный пароль длины y, который совпадает с префиксом правильного пароля, и тратим еще $n-y$ нажатий на то, чтобы дописать пароль до верного.

Сложность решения: $O(n)$.

Задача H. Impossible Puzzle

Можно решить задачу несколькими способами, но здесь приведем только авторское.

Сначала проверим, когда судоку не может существовать — только тогда, когда изначальные два числа равны и они находятся в одной строке, или одном столбце, или в одном квадрате 3 на 3.

Иначе решение судоку всегда существует. Возьмем совершенно любое корректное полностью решенное судоку H (можно скопировать из тестового примера). Затем посмотрим, какие цифры в этом судоку соответствуют двум числам, в почти пустом судоку S, что нам подается на вход. Можно показать, что мы можем привести H к тому, что на местах двух заполненных чисел в S стоят те же числа путем применения некоторого числа следующих операций:

- Выбрать два любых числа различных a и b от 1 до 9 и заменить в H все числа a на b, а все числа b на a;
- Поменять в H местами любые два столбца x и y, если $[\frac{x-1}{3}] = [\frac{y-1}{3}]$;
- Поменять в H местами любые две строки x и y, если $[\frac{x-1}{3}] = [\frac{y-1}{3}]$.

Достаточно не более двух-трех применений каждой операции. Также размер судоку фиксирован, потому сложность решения является постоянной.

Сложность решения: $O(1)$

Задача I. LlrR to Decimal

Задача является реализационной и не требовалось ничего сверх того, что описано в условии. Можно идти по строке и добавлять новые вершины, если символ строки или «L» или «R», переходя в них, иначе, если символ равен «l», то добавляем текущую вершину в стек, а новой текущей вершиной делаем ее левый потомок, если же символ равен «r», то текущей вершиной становится правый потомок, верхней вершины в стеке, которую мы предварительно заберем из стека.

Сложность решения $O(n)$.

Задача J. avt? Что такое avt?

Создадим три переменные $num_a, num_{av}, num_{avt}$. Изначально они равны нулю, для очередного индекса строки они будут показывать, сколько есть до текущего индекса строки включительно различных подпоследовательностей «a» для num_a, «av» для num_{av} и, наконец, «avt» для num_{avt}. Ответ на задачу — значение num_{avt} после разбора последнего индекса строки.

Теперь разберемся, как считать эти переменные в зависимости от очередной буквы строки:

- Если текущий символ равен «a», то $num_a = num_a + 1$;
- Если текущий символ равен «v», то $num_{av} = num_{av} + num_a$;
- Если текущий символ равен «t», то $num_{avt} = num_{avt} + num_{av}$;
- Иначе ничего не происходит.

Сложность решения: $O(n)$.

Задача K. Новогодний вайб

Можно решить задачу за $O(nlogn)$ бинарным поиском с проверкой ответа. Рассмотрим более быстрое и доступное решение: обозначим максимальное возможное значение $k_{max} = \inf$, затем пойдем по уровням елки и будем вычислять, какое максимальное значение может иметь k_i, чтобы текущий уровень елки вписался по размеру в подстриженной красивой елке со значением k_i. Если текущий уровень равен $i > 1$, то $k_i = (a_i - 1)/(i - 1)$. Затем обновляем $k_{max} = \min(k_{max}, k_i)$ и повторяем то же самое со следующим уровнем $i + 1$.

Сложность решения: $O(n)$.

Задача L. Опять 2025

Для того чтобы произведение двух чисел x и y оканчивалось на какое-то число $\overline{abcd}$, например 2025, нужно чтобы произведение некоторого числа последних цифр x и y оканчивалось на это же число $\overline{abcd}$, например 2025.

В нашем случае $x = y$, $\overline{abcd} = 2025$. Можно было перебрать небольшие квадраты чисел и обнаружить, что на 2025 оканчиваются квадраты чисел, которые оканчиваются на 045, 205, 795 и 955. Так, числа 45^2, 205^2, 795^2, 955^2, 1045^2, 1205^2 и т.д. оканчиваются на 2025. Можно заметить, что числа идут четверками — в каждой четверке повторяется окончание числа (045, 205, 795, 955). Идея — сделать бинарный поиск по максимальному числу, которое не больше r, и еще один поиск по максимальному числу, которое меньше l. Для этих чисел мы знаем, какие они по величине, а следовательно можем узнать сколько чисел между l и r.

Сложность решения: $O(log(lr))$.