

Задача А. Пары чисел

Подзадача 1

В первой подзадаче все числа массива неотрицательны, а значит, и наименьшая, и наибольшая достижимая сумма разбиения будут в точности равны сумме всех элементов массива.

Подзадача 2

При $n = 4$ существует всего 3 возможных варианта разбиения:

1. a_1 и a_2 , a_3 и a_4 .
2. a_1 и a_3 , a_2 и a_4 .
3. a_1 и a_4 , a_2 и a_3 .

В данной подзадаче достаточно проверить все три варианта и выбрать два значения — минимальное и максимальное.

Подзадача 3

Все числа в массиве находятся в промежутке от -1 до 1 . Пусть количество -1 в массиве равно x , а количество 1 в массиве равно y . Тогда наименьшая стоимость разбиения достигается при объединении в пары как можно большего числа -1 и 1 . Число таких пар $\min(x, y)$. Наименьшая стоимость в таком случае $x + y - 2 \cdot \min(x, y)$.

Напротив, для достижения наибольшей стоимости разбиения будем объединять в пары сначала -1 и -1 , а также 1 и 1 . Затем начнём формировать пары, в которых один из операндов равен нулю.

Подзадачи 4 и 5

Отсортируем массив a . Функция $|x + y|$ минимизируется, когда x и y выбираются разных знаков, более того, выбирать нужно минимальное и максимальное значение из оставшихся. Минимальной стоимости можно достичь, выбирая пары следующим образом: (a_1, a_n) , (a_2, a_{n-1}) , и т.д.

Для достижения максимальной стоимости разбиения будем также использовать жадную стратегию, выбирая соседние пары чисел. Итоговое разбиение выглядит следующим образом: (a_1, a_2) , (a_3, a_4) , и т.д.

Задача В. Команды

Подзадача 1

Номер каждого игрока состоит только из одной цифры, значит, все игроки будут в разных командах. Ответ n .

Подзадача 2 и подзадача 3

В данных подзадачах можно напрямую перевести каждое число от 1 до n в номер команды и посчитать количество уникальных команд.

Полное решение

Заметим, что количество цифр в номере команды может быть не более 10 , также все цифры идут в строгом порядке. Это означает, что количество команд из k цифр будет равняться количеству сочетаний C_{10}^k , если просуммировать все сочетания для k от 1 до 10 , то в сумме получим 1023 , но команды с номером 0 не может быть, поэтому максимум команд 1022 . Их можно рекурсивно сгенерировать и при генерации новой команды проверять, больше ли номер команды, чем максимальный номер игрока. Если больше, то команда не подходит, и рекурсивную генерацию можно останавливать, иначе эта команда нам подходит и увеличиваем ответ.

Другой вариант — перебрать все подмножества цифр, которые будут входить в номер команды. Зная подмножество цифр, составить наименьший возможный номер не составит труда (нужно брать цифры в порядке возрастания, за исключением случая ведущих нулей) и этот номер нужно сравнить с n .

Отметим так же, что единственный номер команды, который превосходит 10^9 (подзадача 4), это 1023456789 . Таким образом, даже при неэффективной реализации, полное решение можно получить из решения подзадачи 4 при помощи сравнения n с этим числом.

Задача С. Игры с конфетами

Подзадача 1

При $n = 1$ нам всегда придётся играть с одним игроком. Если при этом $a_1 > b_1$, то играть можно несколько раз, пока количество оставшихся конфет будет не меньше a_1 . Несложно посчитать, что количество игр, которые при этом удастся провести, будет равно $k = \lfloor \frac{m-a_1}{a_1-b_1} \rfloor + 1$ (при $b_1 < a_1 \leq m$).

Подзадача 2

Теперь у нас есть выбор, против кого играть и в каком порядке. Отметим, что выгодно сначала играть с игроками, у которых $a_i \leq b_i$ (если такие есть). Далее если остались игроки, у которых $a_i > b_i$, то с ними тоже можно играть несколько раз по аналогии с подзадачей 1, но при этом выгодно сначала играть с тем игроком, у которого разность $a_i - b_i$ минимальна (за одну игру потеряем наименьшее количество конфет).

Подзадача 3

В этой подзадаче с каждым игроком можно сыграть максимум один раз, при этом количество конфет у Васи не меняется. Значит, достаточно найти количество игроков, у которых $a_i \leq m$, эта величина и будет ответом.

Подзадача 4

После каждой игры количество конфет не уменьшается, но игрок обижается и уходит. Отсюда следует, что при возможности провести какую-либо игру, её надо провести, и результат от этого не ухудшится. Тогда достаточно отсортировать всех игроков в порядке возрастания a_i (при равенстве a_i неважно, каковы значения b_i) и каждый раз играть против игрока с наименьшим значением a_i из оставшихся. Если в какой-то момент текущее значение a_i будет больше количества конфет Васи (с учетом уже проведённых игр) или все игроки ушли, то процесс завершается.

Подзадача 5

В общем случае по аналогии с подзадачей 2 имеет смысл сначала играть против соперников с $a_i \leq b_i$. Такие игры можно промоделировать аналогично подзадаче 4. Далее на каждом шаге надо находить игрока с наименьшим значением $a_i - b_i$ (тоже по аналогии с пунктом 2) среди тех, с кем ещё можно играть (то есть $a_i \leq m$). Когда такой игрок найден, с ним нужно провести максимальное количество игр, используя формулу из подзадачи 1. Таким образом, все игры с одним игроком обрабатываются за $O(1)$ и нам может потребоваться максимум n раз находить нового игрока за $O(n)$. Общая сложность решения будет $O(n^2)$.

Подзадача 6

Решение аналогично подзадаче 5, но вместо поиска нового игрока на каждой итерации за $O(n)$, нужно упорядочить всех игроков с параметрами $a_i > b_i$ по возрастанию величины $a_i - b_i$. Тогда можно один раз пройтись по всем игрокам в этом порядке, пропуская игроков, для которых $a_i > m$. Итого, вычислительная сложность решения улучшится до $O(n \cdot \ln(n))$.

Задача D. Василиск Василий

Подзадача 1

Можно перебрать все пары чисел $x, y < 1000$ и для каждой проверить, является ли эта пара сомнительной. Общая сложность работы алгоритма будет $O(10^{2 \cdot n} \cdot n)$.

Подзадача 2

В этой подзадаче можно также воспользоваться перебором, но могут понадобиться неасимптотические оптимизации. Например, можно рассматривать только пары $x < y$ и проверять, что сравнение цифр с конца даст противоположный результат. При этом важно отказаться от перевода чисел в строки и оперировать только цифрами, получаемыми путём взятия остатка при делении на 10.

Подзадача 3

Аналогично можно запустить перебор всех пар, но, в зависимости от эффективности реализации, для получения результата потребуются подождать от нескольких секунд до нескольких минут. Потом посчитанный ответ нужно просто добавить в решение.

Подзадача 4

Для эффективного нахождения ответа потребуется воспользоваться комбинаторикой. В-первых, будем искать количество пар, для которых $x < y$ (потом ответ просто нужно домножить на 2). Далее обратим внимание, что результат сравнения определяется самой левой и самой правой позициями, где два числа различаются. Заметим также, что если числа отличаются только в одной позиции, то такая пара не является сомнительной. Теперь можно перебрать все пары $1 \leq i < j \leq n$, подразумевая, что числа отличаются только на отрезке позиций $[i, j]$. Для каждой такой пары нужно посчитать количество способов расставить цифры. Отметим, что если позиция не является пер-

вой/последней в числе, а мы хотим, чтобы результат сравнения был определённым (например, цифра в первом числе строго больше цифры во втором числе), то у нас есть 45 вариантов. Для крайних позиций в числе — 36 вариантов. Далее, если мы хотим поставить одинаковые цифры (для позиций левее i или правее j), то у нас есть 10 вариантов, за исключением крайних позиций, для которых есть только 9 вариантов. Теперь для фиксированной пары (i, j) во всех позициях вне отрезка $[i, j]$ цифры должны совпадать (9 или 10 вариантов), на позициях i и j результат сравнения определён (36 или 45 вариантов), а для позиций $i < p < j$ цифры могут быть любыми — 100 вариантов на каждую позицию. Теперь для нахождения ответа ans_{ij} числа 10 и 100 нужно возвести в соответствующую степень (до n), что можно сделать за $O(\ln(n))$ для каждой пары или предподсчётом за $O(n)$ в сумме. Итого, сложность решения будет $O(n^2 \cdot \ln(n))$ или $O(n^2)$ в зависимости от реализации.

Подзадача 5

Для полного решения надо заметить, что при фиксированном значении i числа ans_{ij} образуют геометрическую прогрессию (за исключением случая $j = n$, который можно рассмотреть отдельно по аналогии с подзадачей 4). Действительно, при увеличении значения j на 1, мы получим, что у нас стало на 1 разряд больше, в котором цифры могут быть любыми (100 вариантов) и на 1 разряд меньше, где цифры должны быть одинаковыми 10 вариантов. Итого, количество вариантов расставить цифры увеличилось в 10 раз. Тогда сумму значений ans_{ij} для фиксированного i и позиций j : $i < j < n$ можно найти как сумму элементов геометрической прогрессии за $O(\ln(n))$. Итого, общая сложность решения будет $O(n \cdot \ln(n))$.

Задача Е. Странная прокачка скиллов

Подзадача 1

Заметим, что при прокачке скилла блокируется целая ветка от корня до какого-то листа, значит, прокачать можно скиллов не более, чем есть листьев в дереве.

Так как все скиллы равны, то ответом будет количество листов, умноженное на значение силы скилла.

Подзадача 2

Тут все вершины подвязаны к корню. Значит, все вершины, кроме корня — листья. Ответом будет сумма всех положительных листьев, также стоит учесть вариант, что корень может быть больше, чем сила минимального листа, в таком случае к ответу добавляется сила корня.

Подзадача 3

В данной подзадаче любое поддерево — это бамбук, то есть давайте найдем максимальное значение в каждом поддереве (которое начинается в дочерней от корня вершине), и задача сводится к предыдущей подгруппе.

Подзадача 4

В данной подзадаче пройдет полный перебор. Берем незаблокированный скилл с максимальным значением и перебираем все варианты, какую ветку блокировать.

Подзадача 5 и полное решение

Будем решать задачу схожей идеей с подзадачей 3 с отличием в том, что каждое дочернее поддерево это не бамбук. Для каждой дочерней вершины от текущей посчитаем максимум, который можно достигнуть при прокачке скиллов в поддереве с корнем в дочерней вершине. Соответственно, в ответ пойдут все положительные силы, а также вариант, если корень больше нуля и существует поддерево с подсчитанным значением меньше, чем в корне. Это означает, что для каждой вершины нужно поддерживать суммы прокачанных скиллов в каждом поддереве с корнем в дочерней вершине. Также нужно поддерживать последний (ближе всего к текущей вершине) прокачанный скилл, так как при прокачке текущей вершины именно этот скилл будет заменяться.

В зависимости от реализации можно решить за $O(n^2)$ или $O(n)$.

Задача Ф. Хвала Солнцу?

Подзадача 1

В этой подзадаче тело расположено вертикально, поэтому проверить выполнение условий можно просто сравнением координат. Для симметрии двух точек (X_1, Y_1) и (X_2, Y_2) относительно тела должно выполняться: $Y_1 = Y_2$, а координаты X равноудалены от X_A , то есть $X_A - X_1 = X_2 - X_A$

(при $X_1 < X_2$). Для проверки нахождения точки выше/ниже другой точки достаточно сравнить координаты Y .

Подзадача 2

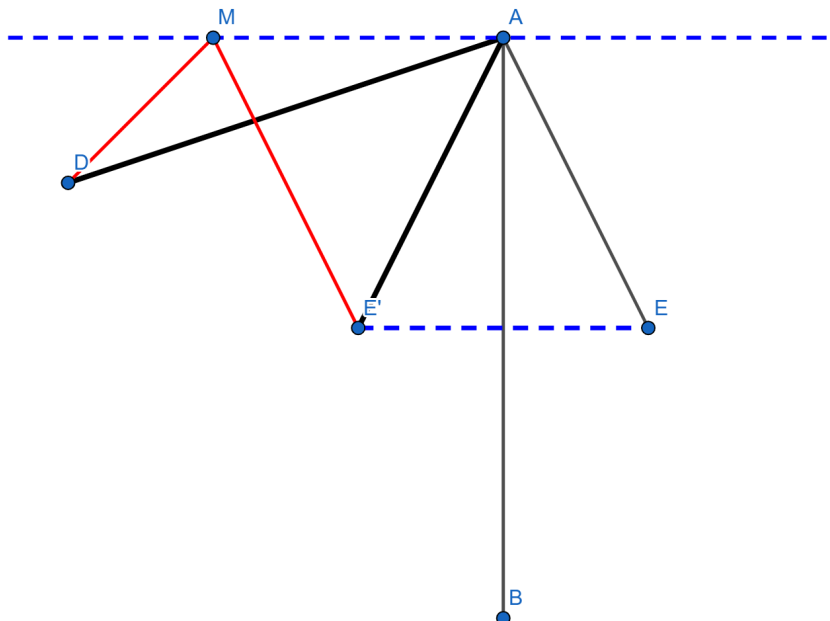
Чтобы проверить условия с учётом поворота, нужно использовать следующие утверждения:

- голова расположена вертикально, если векторное произведение векторов BA и AC равно 0;
- голова расположена выше туловища, если скалярное произведение векторов BA и AC строго больше 0;
- руки расположены симметрично относительно линии туловища, если равны отрезки AE и AD , а также отрезки BE и BD ;
- руки расположены выше туловища, если середина между ними находится выше туловища, проверить это можно по аналогии с головой;
- руки расположены **не** вертикально, если векторное произведение BA и BD **не** равно 0;
- аналогично можно проверить корректность расположения ног.

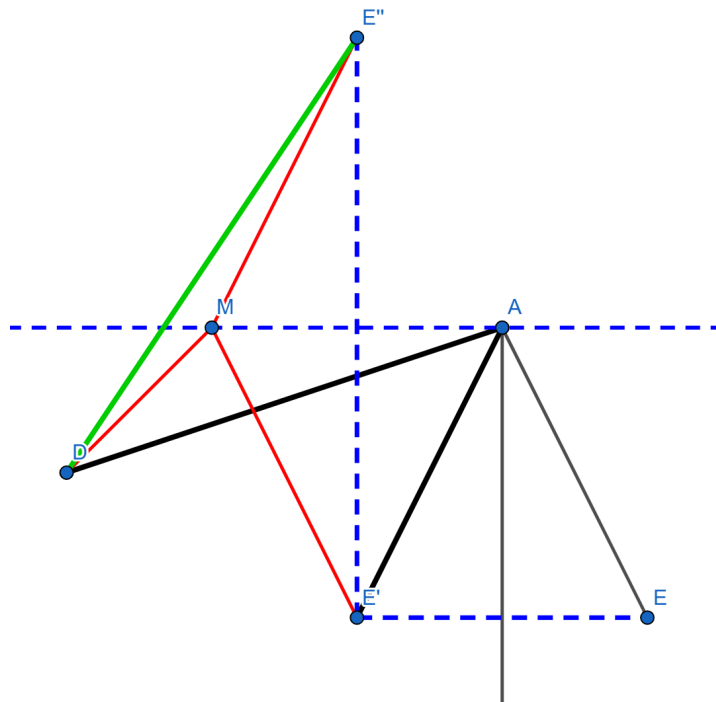
Подзадача 3

Сначала нужно проверить, что голова расположена вертикально и находится выше туловища (по аналогии с подзадачей 2).

Далее нужно независимо решить задачу оптимального перемещения рук и оптимального перемещения ног. Эти задачи независимы и решаются похожим образом — нужно переместить точки, чтобы они находились симметрично относительно прямой и находились выше/ниже заданной точки. Разберём алгоритм на примере рук. Во-первых, отразим одну из рук симметрично относительно туловища (например, получим точку E' путём отражения точки E), тогда задача сведётся к тому, что две руки нужно совместить — переставить в одну точку — причем эта точка должна быть выше туловища. С учетом того, что разрешается ставить руки в нецелые координаты, последнее требование эквивалентно тому, что руки должны быть **не ниже** туловища. Тогда, если хотя бы одна из рук уже не ниже туловища, то нужно вторую руку поставить в эту же точку (меньшим расстоянием всё равно обойтись не удастся). Если же обе точки находятся ниже, то нужно на прямой, ортогональной туловищу и проходящей через верхнюю точку туловища, выбрать некоторую точку M и переставить обе руки туда:



Это классическая математическая задача, она решается довольно просто: нужно отразить точку E' относительно ортогональной прямой, получим точку E'' :



Видно, что длина любой ломаной DME' равна длине ломаной DME'' (красные линии) и эта длина не меньше длины отрезка DE'' (зеленая линия). Тогда расстояние между этими точками (длина зеленой линии) и будет ответом на задачу.

Задача G. Таблица соревнования

Важный момент, который нужно учесть для решения подзадач: если во входных данных встретилось число большее 10^9 , то это число обозначает время сдачи задачи, поскольку с 1970 года прошло более 10^9 секунд, а все другие числа (например, идентификаторы) по условию не превосходят 10^9 .

Подзадача 1

Поскольку участник сдал только одну задачу, то достаточно найти все числа во входных данных, и наибольшее из этих чисел будет соответствовать времени сдачи задачи (это число будет присутствовать дважды, поскольку оно же является последним временем сдачи задачи). Чтобы преобразовать число во время, нужно взять остаток от деления на количество секунд в сутках ($24 * 60 * 60 = 86400$), тогда результат будет равен количеству секунд с начала дня. Преобразовать его в формат $hh : mm : ss$ не составляет труда.

Подзадача 2

Нужно найти имя участника — выделить слово в кавычках, следующее после поля «name». Далее можно найти все времена сдачи задач (числа, превосходящие 10^9), убрать наибольшее из них (это время сдачи последней задачи, оно встречается дважды) и вывести список времён, каждое из которых преобразовано по аналогии с первой подзадачей, предварительно отсортированный по возрастанию.

Подзадача 3

Здесь уже требуется разбить времена сдачи задач по участникам. Сложность заключается в том, что в JSON-формате поля одного объекта могут идти в произвольном порядке, то есть имя участника может идти как до списка задач, так и после. Соответственно, при простом получении списка больших (превосходящих 10^9) чисел в общем случае не получится однозначно определить, к какому участнику они относятся: к имени, которое указано до списка, или к имени, которое указано после списка. Для решения этой проблемы нужно учитывать вложенность объектов: существует объект, в котором находятся все времена, относящиеся к одному участнику (и только они), а также имя этого участника. Тут возникает другая проблема — неизвестно, какова вложенность d данного

объекта. Но с учётом отсутствия штрафов за неверные послылки, наиболее эффективный способ: отправлять решения с разными значениями необходимой (но неизвестной) вложенности объекта. В тесте 3 одним из подходящих значений этой вложенности было $d = 3$. При фиксированном d нужно было поддерживать текущую вложенность при обработке входных данных: символ '{' увеличивает вложенность на 1, а символ '}' — уменьшает на 1. Когда вложенность стала меньше d , мы считаем, что данные по текущему участнику закончены, нужно взять последнее найденное имя и поставить ему в соответствие все времена, полученные после предыдущего объекта, после чего очистить список времен.