Задача B. Грубый баскетбол

В 1-й подзадаче **(25 баллов)** можно перебрать количество игроков в команде, которые набрали 5 фолов. Обозначим это число за k. Во первых, k не может быть больше $\frac{s}{5}$. Во вторых, оставшиеся $n-k$ человек должны набрать не более четырёх фолов каждый. То есть, $\frac{s-5k}{n-k} \leqslant 4$, или $s-5k \leqslant 4(n-k)$. Из тех k, у которых выполнены эти условия, выберем минимальное и максимальное.

Во 2-й подзадаче **(35 баллов)** перебор уже не проходит ограничения по времени. Но из неравенств выше можно получить следующие ограничения на k: $k \leqslant \frac{s}{5}$ и $s-4n \leqslant k$. Отсюда получаем формулу для максимума — $\lfloor \frac{s}{5} \rfloor$, и для минимума — $max(0, s-4n)$.

В 3-й подзадаче **(40 баллов)** нужно сделать все то же самое, но не забыть про большие числа. Например, в C++ это тип `long long`.

Задача C. Химические растворы

В 1-й подзадаче **(25 баллов)** можно вручную рассмотреть все возможные комбинации сосудов, всего таких комбинаций 7. Из них выбрать нужные для минимальной и максимальной концентрации.

Во 2-й подзадаче **(35 баллов)** можно перебрать все возможные комбинации сосудов с помощью перебора всех масок. Это делается за $O(n \cdot 2^n)$. Из них выбрать нужные для минимальной и максимальной концентрации.

В 3-й подзадаче **(40 баллов)** нужно заметить следующее. Допустим, что мы выбрали какие-то k сосудов. Тогда для их концентрации будет верно следующее неравенство:

$$min(\frac{s_i}{a_i}) \leqslant \frac{s_1 + s_2 + \ldots + s_k}{a_1 + a_2 + \ldots + a_k} \leqslant max(\frac{s_i}{a_i}),$$

где $min(\frac{s_i}{a_i})$ и $max(\frac{s_i}{a_i})$ — это сосуды с минимальной и максимальной концентрацией соответственно. Тогда нам достаточно просто из всех сосудов найти сосуд с минимальной и максимальной концентрацией и вывести их в ответ.

Задача D. Any Percent

Выведем формулу, по которой можно получить ответ.

Если удар второго типа наносит не меньше урона, чем удар первого типа, то удар первого типа применять бессмысленно. В таком случае игрок гарантированно выигрывает за $\lceil \frac{X}{B} \rceil$ ударов, где $\lceil v \rceil$ означает число v, округленное вверх.

Иначе, заметим, что удары силы A эффективней, и из жадных соображений нужно применить их как можно больше. $Y-1$ удар первого типа можно совершить в любой момент, но можно также совершить еще один последним. В таком случае, можно изначально «зарезервировать» $K = \min\left(\lceil \frac{X}{A} \rceil, Y\right)$ последних ударов, и если их недостаточно, добить босса за $\lceil \frac{X-K \cdot A}{B} \rceil$ ударов второго типа.

Задача E. Баня

В первой подзадаче **(30 баллов)** нужно заметить, что если $f_i \leqslant N+1$, то достаточно уговорить $f_i - 1$ человек, чтобы собрались все участники. Иначе, нужно уговорить каждого из N участников отдельно.

Для решения второй подзадачи **(30 баллов)** необходимо отметить, что уговаривать выгоднее самых *требовательных* людей, то есть людей с наибольшим f_i. Чтобы это понять, заметим, что в любой момент уговор человека с большим f_i, чем другого, не ухудшает ответ, а в некоторых случаях даже улучшает. Решение выглядит следующим образом: на каждом этапе нужно найти всех людей, которые уже согласны идти в баню. Если остались ещё люди, нужно найти максимум в массиве f_i и удалить его, уговорив человека напрямую. Потом продолжать работу алгоритма, пока массив f_i не станет пустым. Время работы алгоритма — $O(N^2)$.

Однако, заметим, что присоединяющиеся без уговоров люди идут по возрастанию значений f_i. Тогда, отсортировав массив по возрастанию, можно использовать два указателя, чтобы ускорить работу алгоритма. Первый указатель будет идти от начала массива в конец и будет отвечать за людей, которые присоединяются к компании без уговоров. А второй будет идти от конца массива в начало и будет отвечать за участников сборов, которых нужно уговорить. Тогда, поддерживая текущий размер компании, нужно вначале обработать всех людей из начала массива, затем добавить

одного человека из конца массива, снова обработать людей из начала массива, и так далее. Остановить алгоритм нужно, когда два указателя встретятся. Таким образом решается третья подзадача **(40 баллов)** за $O(N \cdot log(N))$.

Задача F. Урок физкультуры

Для решения первой подзадачи **(15 баллов)** нужно заметить, что при $K = 2$ все ученики на нечётных позициях произнесут «Первый», а на чётных — «Второй». Значит, достаточно пройтись по каждому из N человек, проверить чётность его позиции и добавить соответствующее значение в общую сумму. При $K = 1$, очевидно, ответ равен N.

Во второй подзадаче **(20 баллов)** нужно также пройтись по каждому из N человек и посчитать, что будет говорить каждый школьник. Первый скажет «Первый», а каждый следующей произнесёт число на 1 больше. Но если школьник произносит «K-й», то следующий за ним крикнет «Первый».

В третьей подзадаче **(30 баллов)**, как и в первой, отметим, все ученики на нечётных позициях произнесут «Первый», а на чётных — «Второй». На чётных позициях стоят N div 2 учеников, на нечётных — $N - (N \text{ div } 2)$. Поэтому общая сумма равна $(N \text{ div } 2) \cdot 2 + N - (N \text{ div } 2)$, что равно $N + N \text{ div } 2$. Здесь и далее div означает деление нацело.

Для решения последней подзадачи **(35 баллов)** нужно заметить, что числа, названные детьми, разбиваются на арифметические прогрессии длины K. Сумма арифметической прогрессии из x чисел от 1 до x равна $\frac{x \cdot (x+1)}{2}$. Все названные числа разобьются на $N \text{ div } K$ таких прогрессий от 1 до K и ещё одну от 1 до $N \bmod K$ (если N не делится на K). Общий ответ равен

$$(N \text{ div } K) \cdot \frac{K \cdot (K+1)}{2} + \frac{(N \bmod K) \cdot ((N \bmod K) + 1)}{2}.$$

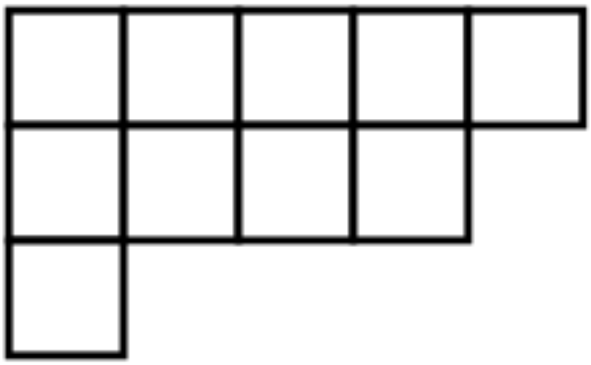
Задача G. Пары по алгоритмам

В первой подзадаче **(30 баллов)** нужно просто K раз применить преобразование. Оно делается таким образом: сначала нужно записать количество чисел, больших либо равных 1 (их ровно столько, сколько всего выписанных на доске чисел). Затем количество чисел, больших либо равных 2, затем больших либо равных 3, и так далее. Раз работа ведётся с отсортированным по возрастанию набором, можно просто поддерживать указатель на первый элемент, больший либо равный рассматриваемого числа. Так, если нас интересует количество чисел, больших либо равных K, и первое такое число — это A_i, то искомое количество равно $N - i + 1$. Указатель движется только вправо, и после прохождения по всему набору новые числа нужно отсортировать. Чтобы оценить время работы алгоритма, нужно заметить, что размер массива и наибольший элемент никогда не превзойдут $max(N, max(A_i))$. Если $C = max(N, max(A_i))$, то алгоритм работает за $O(K \cdot C \cdot log(C))$.

Если же заметить, что после каждого преобразования новый набор оказывается невозрастающим, то вместо сортировки его можно просто развернуть. Это решает вторую подзадачу **(20 баллов)** за $O(K \cdot C)$.

Для решения последней подзадачи **(50 баллов)** нужно заметить интересный факт: если применить преобразование к любому набору 2 раза, то результатом будет исходный набор (доказательство ниже). Это означает, что если $K \mod 2 = 0$, то преобразование можно не совершать, а достаточно вывести исходный набор. А если $K \mod 2 = 1$, то достаточно выполнить ровно одно преобразование и вывести полученный набор.

Доказать, что два преобразования подряд не меняют набор, можно с помощью диаграммы Юнга. Диаграмма Юнга — это фигура из клеток, в которой размеры столбцов соответствуют невозрастающему набору чисел. Так, если на доска написаны числа $1, 2, 2, 2, 3$, можно их развернуть и построить следующую диаграмму:



Но если диаграмму записать по столбцам, а прочитать по строчкам, то мы получим как раз преобразованный массив. В этом случае, записав длины строк снизу вверх, получим $1, 4, 5$ — это и есть массив после одного преобразования. Получается, когда мы выполняем преобразование, мы отражаем диаграмму Юнга относительно главной диагонали. Два преобразования подряд выполняют два отражения, в результате которых диаграмма возвращается к исходному положению. А значит, набор чисел остаётся прежним.

Задача N. Баба Марья

В первой подзадаче **(20 баллов)** если есть квартира ширины 2, то можно распылить аромат в ней. Тогда запах заполнит все квартиры верху и снизу, а значит, заполнит весь дом. Если же все квартиры имеют ширину 1, то где бы мы не распылили запах, он заполнит только половину дома. Значит, ответ равен либо Q, либо M.

Вторую подзадачу **(20 баллов)** можно решить методом двух указателей. Попробуем сначала распылить аромат в первой квартире первого этажа. Будем поддерживать указатель на самую правую квартиру первого этажа, в которую дошёл запах; и самую правую квартиру второго этажа, в которую дошёл запах. Если правые концы этих двух квартир не совпадают, то из той квартиры, которая заканчивается правее, можно ещё распространить запах на другой этаж. Таким образом, можно сдвинуть указатель, который соответствует квартире с более левым концом. Если текущие квартиры на обоих этажах имеют одну и ту же правую границу, то запах через них не проберётся. Посчитаем суммарное количество пройденных квартир и обновим ответ.

Чтобы завершить решение, нужно запустить этот процесс несколько раз. Когда процесс завершится на какой-то паре квартир, просто сдвинем указатели на одну позицию вправо, и попытаемся распылить аромат там. Максимальное количество квартир среди всех запусков и будет ответом.

Для решения третьей подзадачи **(25 баллов)** представим дом поле в виде клетчатого поля $N \times M$, где каждая клетка соответствует одному окну. Построим на этом поле граф, где вершина — это окно, а ребро соответствует распространению запаха. То есть, во всех окнах кроме граничных будет ребро вверх и вниз; а рёбра влево и вправо будут между окнами одной квартиры. С помощью поиска в глубину определим компоненты связности графа и посчитаем в каждой компоненте количество квартир. Наибольшее количество квартир и будет ответом. Такое решение работает за $O(N \cdot M)$.

Для решения четвёртой подзадачи **(35 баллов)** нужно также построить граф, но на квартирах, а не на окнах. Поскольку $N \cdot M$ может быть очень большим, а количество квартир Q ограничено, можно создать граф на Q вершинах, и соединить рёбрами квартиры, между которыми возможно перемещение запаха. Чтобы это сделать, воспользуемся идеей второй подзадачи: методом двух указателей. Понятно, что рёбра будут проводиться только между квартирами соседних этажей. Поэтому можно рассмотреть каждую пару соседних этажей и запустить процесс, аналогичный второй подзадаче. Каждый раз, когда мы сдвигаем указатель, будем проводить ребро. Таким образом, если на i-м этаже k_i квартир, а на $i+1$-м этаже k_{i+1} квартир, то между ними будет проведено не более $k_i + k_{i+1}$ рёбер. Можно посчитать, что суммарно такой граф будет иметь не более $2 \cdot Q$ рёбер. Наконец, запустим поиск в глубину или в ширину, как в третьей подзадаче, и получим решение за $O(M + Q)$.

Альтернативное решение: посмотрим, как выглядит множество квартир, затронутое запахом. Если представить дом как клетчатое поле $N \times M$ (как в третьей подзадаче), то запах распылился

на каком-то интервале столбцов, и каждый столбец был покрыт полностью. Таким образом, запах всегда распыляется на каком-то прямоугольнике. Нижняя и верхняя граница прямоугольника — это всегда нижняя и верхняя граница дома. А вот левая и правая граница — не обязательно. Но если левая или правая граница не является границей дома, то она должна быть левой или правой границей всех M квартир этого столбца. Таким образом, если посчитать все координаты, где встречается граница квартиры, и на какой-то вертикальной линии встретится M границ, то эта линия является границей области распыления. Остаётся только посчитать ответ — между каждыми позициями, где одновременно встречается M границ квартир, посчитать количество квартир. Это можно сделать методом сканирующей прямой или просто сортировкой.

Обратите внимание, что в этой задаче большой размер ввода. Рекомендуется использовать более быстрые способы чтения данных. Например, на языке `Python` это можно сделать следующим образом:

```
import sys
lines = sys.stdin.readlines()
```