

<https://codeforces.com/contest/1482/problem/A>

А. Побег

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Майкл обвиняется в нарушении масочного режима и создании риска распространения коронавируса. Теперь он приговорён отбывать наказание в тюрьме. К счастью, Майкл знает внутреннее устройство тюрьмы, а оно довольно простое.

Тюрьма может быть представлена в виде прямоугольника $a \times b$, разбитого на ab клеток. Каждая клетка обозначает тюремную камеру, перегородки между клетками обозначают стены между камерами, и внешние перегородки обозначают внешние стены тюрьмы. Перед заключением Майкл может воспользоваться своими связями среди сотрудников тюрьмы, чтобы сделать (очень хорошо спрятанные) отверстия в стенах (как во внутренних, так и во внешних). Майкл не знает, в какую камеру он попадёт, но хочет, чтобы вне зависимости от этого он потом смог выбраться наружу. С другой стороны, чтобы обезопасить себя, он хочет сломать как можно меньше стен.

Найдите наименьшее количество стен, которые нужно сломать, чтобы после этого из каждой камеры можно было выбраться наружу.

Входные данные

В первой строке записано единственное число t ($1 \leq t \leq 100$) — количество наборов входных данных.

В каждой из следующих t строк записано по два целых числа a and b ($1 \leq a, b \leq 100$), задающих очередной набор входных данных.

Выходные данные

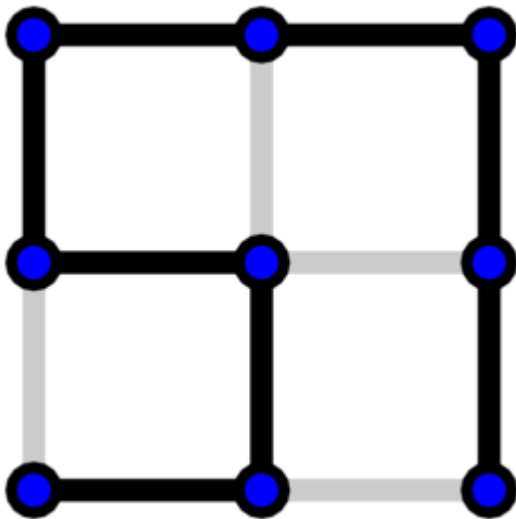
Для каждого набора входных данных в отдельной строке выведите единственное число — ответ на задачу.

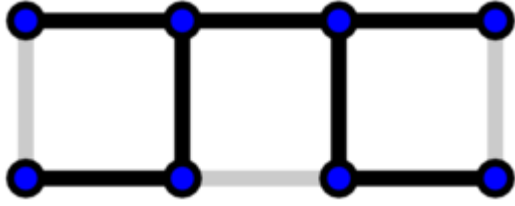
Пример

входные данные	Скопировать
2 2 2 1 3	
выходные данные	Скопировать
4 3	

Примечание

Возможные планы побега для наборов входных данных из примера изображены ниже. Серым цветом обозначены сломанные стены, черным — не сломанные.





Разбор задачи A

1482A - Побег

Каждая сломанная стена увеличивает число связных областей плоскости максимум на 1. Поскольку в конце каждая клетка должна быть в той же области, что и внешнее пространство, в конце должна быть только одна связная область, а значит, ответ хотя бы nm (потому что вначале областей было $nm + 1$).

Добиться цели, удалив ровно nm стен, можно, например, сломав верхнюю стену в каждой клетке.

<https://codeforces.com/contest/1482/problem/B>

В. Восстановление модуля

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

За первое место на олимпиаде Лёше подарили много массивов целых чисел, уверив его, что эти массивы очень дорогие. Сразу после награждения Лёша принял решение продать их. На двери пункта приема массивов было сказано, что принимаются только массивы, которые можно сжать в генератор, работающий по следующему принципу:

На вход генератору дается четверка целых чисел n, m, c, s , при этом n и m положительны, s неотрицательно, а для c верно, что $0 \leq c < m$. Массив a из n элементов генерируется по следующим правилам:

- $a_1 = s \bmod m$, где $x \bmod y$ обозначает остаток от деления x на y ;
- $a_i = (a_{i-1} + c) \bmod m$ для всех i таких, что $1 < i \leq n$.

Например, если $n = 5, m = 7, c = 4$, а $s = 10$, то $a = [3, 0, 4, 1, 5]$.

Цена такого массива вычисляется очень просто — это значение m в этой четверке чисел.

Лёше стало интересно, сколько денег он может потребовать за каждый свой массив. Помогите Лёше определить для каждого массива, можно ли его задать какой-нибудь четверкой чисел n, m, c, s , и, если да, найти максимальное значение m среди всех подходящих четверок.

Входные данные

На первой строке находится целое число t ($1 \leq t \leq 10^5$) — количество массивов.

Первая строка описания массива содержит целое число n ($1 \leq n \leq 10^5$) — размер очередного массива, полученного Лешей.

Вторая строка описания массива содержит n целых чисел a_1, a_2, \dots, a_n ($0 \leq a_i \leq 10^9$) — элементы массива.

Гарантируется, что суммарный размер массивов не превосходит 10^5 .

Выходные данные

Для каждого массива выведите:

- -1 , если нет такой четверки чисел, что генерирует этот массив;
- 0 , если есть четверки чисел со сколь угодно большим m ;
- максимальное значение m и любое подходящее c ($0 \leq c < m$) в остальных случаях.

Пример

входные данные

Скопировать

```
6
6
1 9 17 6 14 3
3
4 2 2
3
7 3 4
3
2 2 4
5
0 1000000000 0 1000000000 0
2
1 1
```

выходные данные

Скопировать

```
19 8
-1
-1
-1
2000000000 1000000000
0
```

Разбор задачи В

1482В - Восстановление модуля

Давайте отдельно разберем случай $c == 0$. Для этого проверим, что для любого i такого, что $1 \leq i < n$ верно, что $arr[i] == arr[i + 1]$. Если это так, то модуль может быть бесконечно большим.

Если это не так и есть пара $arr[i] == arr[i + 1]$, значит с одной стороны, $(x + c) \% mod = x$, а значит $c = 0$, но этот случай мы уже отсекали. Значит ответ -1 , ведь решения нет.

Теперь посмотрим на то, что $(x + c) \% mod$ может изменить x либо на $x + c$, либо на $x - (mod - c)$.

Посмотрим на все положительные дельты. Они должны быть одинаковыми. Аналогично и с отрицательными.

Если нет отрицательно или положительной дельты - модуль может быть сколь угодно большим. В противном случае модуль равен сумме модулей положительной и отрицательной дельт. Осталось только проверить, что он подходит.

<https://codeforces.com/contest/1482/problem/C>

С. Базовая дипломатия

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

У Алексея есть n друзей. А ещё Алексей сейчас в отпуске, поэтому у него есть целых m дней, чтобы поиграть в эту новую популярную кооперативную игру! Поскольку эта игра кооперативная, Алексею нужен один сокомандник в каждый из m дней.

Каждый день какие-то его друзья свободны и готовы играть в эту игру, а остальные заняты и не могут. Каждый день Алексей должен выбрать одного из свободных друзей и предложить ему поиграть (а тот, разумеется, согласится). Однако если какой-то из друзей будет играть с Алексеем суммарно строго больше $\left\lceil \frac{m}{2} \right\rceil$ раз (округление вверх), то все остальные обидятся. Конечно же, Алексей не хочет никого обижать.

Помогите ему выбрать сокомандников таким образом, чтобы никто не играл с Алексеем суммарно строго больше $\left\lceil \frac{m}{2} \right\rceil$ раз.

Входные данные

Во входных данных находятся несколько наборов входных данных. В первой строке находится одно целое число t ($1 \leq t \leq 10\,000$) — количество наборов входных данных. Далее следуют наборы входных данных.

Первая строка каждого набора входных данных содержит два натуральных числа n и m ($1 \leq n, m \leq 100\,000$) — количество друзей и количество дней, соответственно.

В i -й из следующих m строк содержится натуральное число k_i ($1 \leq k_i \leq n$), за которым следуют k_i попарно различных целых чисел f_{i1}, \dots, f_{ik_i} ($1 \leq f_{ij} \leq n$), разделённых пробелами — свободные друзья в день i .

Гарантируется, что сумма значений n и сумма значений m по всем наборам входных данных не превосходят 100 000. Гарантируется, что сумма всех k_i по всем дням всех наборов входных данных не превосходит 200 000.

Выходные данные

Выведите ответ для каждого набора входных данных. Если невозможно никого не обидеть, то выведите «NO».

В противном случае в первой строке выведите «YES», а во второй строке выведите m целых чисел c_1, \dots, c_m , разделённых пробелами. Каждое c_i должно быть номером друга, взятого в команду в i -й день (соответственно, это должно быть одно из чисел f_{ij}).

Никакое значение не должно встречаться больше, чем $\left\lceil \frac{m}{2} \right\rceil$ раз. Если есть несколько возможных ответов, выведите любой из них.

Пример

входные данные	Скопировать
<pre>2 4 6 1 1 2 1 2 3 1 2 3 4 1 2 3 4 2 2 3 1 3 2 2 1 1 1 1</pre>	
выходные данные	Скопировать
<pre>YES 1 2 1 1 2 3 NO</pre>	

Разбор задачи С

1482C - Базовая дипломатия

Сначала для каждого дня произвольно выберем друга из списка. Заметим, что при таком выборе максимум один друг будет играть больше $\left\lceil \frac{m}{2} \right\rceil$ раз. Назовем его f . Теперь мы хотим немного подправить расписание так, что f играет ровно $\left\lceil \frac{m}{2} \right\rceil$ раз. Это будет гарантировать корректность нашего ответа. Для этого просто пройдемся по всем дням и, если в некоторый день назначен f и может играть кто-то другой, назначим на этот день кого угодно, кроме f . Такие замены будем производить пока f играет больше $\left\lceil \frac{m}{2} \right\rceil$ раз. Существует только один случай, когда добиться этого невозможно: если в больше чем $\left\lceil \frac{m}{2} \right\rceil$ дней f является единственным другом, который может играть.

<https://codeforces.com/contest/1482/problem/D>

D. Плейлист

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

У Аркадия есть плейлист, в котором изначально n песен, пронумерованных от 1 до n в том порядке, в котором они находятся в плейлисте. Аркадий начинает слушать песни из плейлиста друг за другом, начиная с песни 1. Плейлист зациклен, то есть после того, как он прослушает последнюю песню, он опять начнет слушать с начала плейлиста.

У каждой песни есть жанр, определяемый натуральным числом a_i . Пусть Аркадий только что прослушал песню с жанром y , а до нее — песню с жанром x . Если $\gcd(x, y) = 1$, он думает, что песни не сочетаются, и удаляет последнюю прослушанную песню (с жанром y) из плейлиста. После этого он продолжает слушать песни как обычно, пропуская уже удаленные, при этом он **забывает**, что слушал какие-то песни до удаления. Иными словами, он не может удалить песню сразу после того, как удалил предыдущую. Здесь $\gcd(x, y)$ обозначает **наибольший общий делитель (НОД)** чисел x и y .

Например, если жанры песен в начальном плейлисте равны $[5, 9, 2, 10, 15]$, то прослушивание происходит следующим образом: $[5, 9, 2, 10, 15] \rightarrow [5, 9, 2, 10, 15] \rightarrow [5, 2, 10, 15]$ (т. к. $\gcd(5, 9) = 1$) $\rightarrow [5, 2, 10, 15] \rightarrow [5, 2, 10, 15] \rightarrow [5, 2, 10, 15] \rightarrow [5, 2, 10, 15] \rightarrow [5, 2, 10, 15] \rightarrow [5, 2, 10, 15] \rightarrow [5, 10, 15]$ (т. к. $\gcd(5, 2) = 1$) $\rightarrow [5, 10, 15] \rightarrow [5, 10, 15] \rightarrow \dots$ Жирным выделены последние две прослушанные песни. Обратите внимание, после удаления некоторой песни Аркадий забывает, что слушал ее и предыдущую.

Вам дан начальный плейлист, определите, какие песни будут в итоге удалены и в каком порядке.

Входные данные

Во входных данных находятся несколько наборов входных данных. В первой строке находится одно целое число t ($1 \leq t \leq 10\,000$) — количество наборов входных данных. Далее следуют наборы входных данных.

Первая строка набора входных данных содержит целое число n ($1 \leq n \leq 10^5$) — количество песен.

Вторая строка содержит n целых чисел a_1, a_2, \dots, a_n ($1 \leq a_i \leq 10^9$) — жанры песен.

Гарантируется, что сумма значений n по всем наборам входных данных не превосходит 10^5 .

Выходные данные

Для каждого набора входных данных выведите одну строку. Сначала выведите одно целое число k — количество песен, которые будут удалены. Затем выведите k различных целых чисел — номера удаленных песен в порядке удаления.

Пример

ВХОДНЫЕ ДАННЫЕ	Скопировать
5 5 5 9 2 10 15 6 1 2 4 2 4 2 2 1 2 1 1 1 1 2	
ВЫХОДНЫЕ ДАННЫЕ	Скопировать
2 2 3 2 2 1 2 2 1 1 1 0	

Примечание

Первый набор входных данных объяснен в условии.

Во втором наборе плейлист изменяется следующим образом: $[1, 2, 4, 2, 4, 2] \rightarrow [1, 2, 4, 2, 4, 2] \rightarrow [1, 4, 2, 4, 2]$ (т. к. $\gcd(1, 2) = 1$) $\rightarrow [1, 4, 2, 4, 2] \rightarrow [1, 4, 2, 4, 2] \rightarrow [1, 4, 2, 4, 2] \rightarrow [1, 4, 2, 4, 2] \rightarrow [1, 4, 2, 4, 2] \rightarrow [4, 2, 4, 2]$ (т. к. $\gcd(2, 1) = 1$) $\rightarrow [4, 2, 4, 2] \rightarrow \dots$

Во третьем наборе плейлист изменяется следующим образом: $[1, 2] \rightarrow [1, 2] \rightarrow [1]$ (т. к. $\gcd(1, 2) = 1$) $\rightarrow [1] \rightarrow [1]$ (Аркадий послушал одну и ту же песню подряд дважды) $\rightarrow []$ (т. к. $\gcd(1, 1) = 1$).

Четвертый набор входных данных совпадает с предыдущим после удаления второй песни.

В пятом примере Аркадий слушает одну и ту же песню снова и снова, но т. к. $\gcd(2, 2) \neq 1$, он не удаляет ее.

Разбор задачи D

1482D - Плейлист

Будем называть пару песен *плохой*, если НОД их жанров равен 1. Для двух данных песен это легко проверить за $O(\log C)$, используя алгоритм Евклида.

Так как удалений будет не более n , то мы можем просто симулировать процесс, и это уложится в ограничение по времени, если мы сможем быстро находить песню, которая будет удалена следующей. Будем поддерживать упорядоченное множество песен в плейлисте, а также другое упорядоченное множество плохих пар соседних песен в плейлисте. Легко понять, что после того, как мы удалили песню из плейлиста, в этих множествах нужно сделать лишь константное число изменений. Пусть мы удалили песню b , и в этот момент предыдущая песня в плейлисте была a , а следующая песня — c . Индексы a и c легко найти, используя первое из поддерживаемых множеств. Тогда нам нужно:

- удалить b из множества, в котором хранится плейлист;
- удалить (a, b) из множества плохих пар;
- удалить (b, c) из множества плохих пар, если это плохая пара;
- добавить (a, c) в множество плохих пар, если это плохая пара.

Затем можно перейти к следующей песне, которую нужно удалить. Это легко сделать, используя множество плохих пар. Если мы будем поддерживать данные множества в достаточно быстрой структуре данных (например, сбалансированном дереве поиска, стандартного `set` в языке C++ достаточно), то мы получим достаточно быстрое время работы ($O(n(\log n + \log C))$).

Также эту задачу можно решить с помощью двусвязных списков или CHM, тогда асимптотика будет $O(n \log C)$.

<https://codeforces.com/contest/1482/problem/E>

Е. Панорама города

ограничение по времени на тест: 2.5 секунд
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Алиса в Нью-Йорке! Она хочет сделать несколько снимков знаменитых небоскребов и отправить набор фотографий Бобу. Алиса хочет составить наиболее красивый набор фотографий, и в этом ей требуется ваша помощь.

В городе n зданий, расположенных в линию слева направо, i -е из них имеет положительную высоту h_i . Высоты всех n зданий в городе различны. Кроме того, у каждого здания есть параметр красоты b_i . Красота зданий может быть положительна или отрицательна (или равна нулю), потому что некоторые здания портят вид.

Набор фотографий состоит из одной или более фотографий зданий. На каждом фото должно быть изображено одно или несколько зданий, расположенных подряд. Каждое здание должно попасть на **ровно одно** фото. Другими словами, если есть здание, которое не попало ни на одно фото, или присутствует более чем на одном фото, то такой набор фотографий считается некорректным.

Красота одной фотографии равняется красоте b_i самого низкого здания на фотографии. Общая красота набора фотографий равна сумме красот всех фотографий в наборе. Помогите Алисе определить максимально возможную красоту корректного набора фотографий.

Входные данные

Первая строка содержит одно целое число n ($1 \leq n \leq 3 \cdot 10^5$) — количество зданий.

Вторая строка содержит n различных целых чисел h_1, h_2, \dots, h_n ($1 \leq h_i \leq n$), где i -е число равно высоте i -го здания.

Третья строка содержит n целых чисел b_1, b_2, \dots, b_n ($-10^9 \leq b_i \leq 10^9$), где i -е число равно красоте i -го здания.

Выходные данные

Выведите одно число — максимальную красоту, которую может иметь корректный набор фотографий.

Примеры

входные данные	Скопировать
5 1 2 3 5 4 1 5 3 2 4	
выходные данные	Скопировать
15	
входные данные	Скопировать
5 1 4 3 2 5 -3 4 -10 2 7	
выходные данные	Скопировать
10	
входные данные	Скопировать
2 2 1 -2 -3	
выходные данные	Скопировать
-3	
входные данные	Скопировать
10 4 7 3 2 5 1 9 10 6 8 -4 40 -46 -8 -16 4 -10 41 12 3	
выходные данные	Скопировать
96	

Примечание

В первом примере максимальная красота достигается, если Алиса сделает пять фотографий, каждая из которых содержит одно здание.

Во втором примере Алиса может достичь максимальной красоты набора, равной 10, если сделает четыре фотографии: три содержат одно здание каждая (здания 1, 2 и 5), их красоты равны -3, 4 и 7 соответственно, и еще одна фотография содержит здания 3 и 4, ее красота равна 2.

В третьем примере Алиса может просто сделать одно фото всего города.

В четвертом примере Алиса может сделать следующие фотографии для достижения максимальной красоты набора: отдельно фотографии зданий 1, 2, 8, 9 и 10, а также одно фото зданий 3, 4, 5, 6 и 7 вместе.

Разбор задачи E

1482E - Панорама города

Будем решать задачу методом динамического программирования. Наивный алгоритм за $O(n^2)$ выглядит следующим образом: обозначим через dp_i максимальную красоту, которую можно получить на подотрезке от 1 до i . Проверим все возможные варианты для последнего отрезка: $dp_i = \max_{j \leq i} (dp_{j-1} + b_{j..i})$. Осталось ускорить это решение.

Предположим, в данный момент мы считаем dp_i . Заметим, что если j — позиция ближайшего меньшего числа слева от i (то есть $h_j < h_i$, и для всех $j < j' < i$ выполнено $h_{j'} > h_i$), и если оно попадет на одно фото со зданием i , то в таком случае наибольшее значение, которое может принять dp_i — это dp_j , поскольку красота последнего фото будет определяться зданием, которое находится не правее j .

Осталось лишь проверить все позиции от $j + 1$ до i в качестве левого здания на последнем фото. Но мы знаем, что все здания между j и i выше h_i , поэтому красота последнего фото в таком случае будет определяться зданием i , так что ответ в таком случае будет равен $dp_{k-1} + b_i$, где k — выбранная позиция. Чтобы найти это значение, нам нужна структура, поддерживающая максимум на отрезке и обновляющая значение в точке, например, дерево отрезков. Итоговое решение имеет сложность $O(n \log n)$.

Также можно пройти по зданиям слева направо, поддерживая стек возрастающих зданий, чтобы всегда знать, где находится ближайшее менее высокое здание, и заодно поддерживать оптимальные значения dp , соответствующие каждому зданию из стека, что даёт линейное по времени решение, но это было не обязательно.

<https://codeforces.com/contest/1482/problem/E>

Ф. Полезные рёбра

ограничение по времени на тест: 5 секунд
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Дан взвешенный неориентированный граф на n вершинах, а также q троек (u, v, l) , в каждой из которых u и v — некоторые вершины графа, а l — натуральное число. Назовём ребро e *полезным*, если существует хотя бы одна тройка (u, v, l) , для которой найдётся путь (не обязательно простой) в графе со следующими свойствами:

- u и v являются концами этого пути,
- путь содержит ребро e ,
- сумма весов рёбер этого пути не превосходит l .

Найдите количество полезных рёбер в этом графе.

Входные данные

В первой строке даны два целых числа n и m ($2 \leq n \leq 600$, $0 \leq m \leq \frac{n(n-1)}{2}$).

Каждая из следующих m строк содержит три числа u, v и w ($1 \leq u, v \leq n$, $u \neq v$, $1 \leq w \leq 10^9$), обозначающих, что в графе есть ребро между вершинами u и v с весом w .

Следующая строка содержит единственное число q ($1 \leq q \leq \frac{n(n-1)}{2}$) — количество троек.

Каждая из следующих q строк содержит по три числа u, v и l ($1 \leq u, v \leq n$, $u \neq v$, $1 \leq l \leq 10^9$), обозначающих тройку (u, v, l) .

Гарантируется, что:

- граф не содержит петель и кратных рёбер,
- все пары (u, v) в тройках попарно различны.

Выходные данные

Выведите одно число — количество полезных рёбер в графе.

Примеры

входные данные	Скопировать
4 6 1 2 1 2 3 1 3 4 1 1 3 3 2 4 3 1 4 5 1 1 4 4	
выходные данные	Скопировать
5	
входные данные	Скопировать
4 2 1 2 10 3 4 10 6 1 2 11 1 3 11 1 4 11 2 3 11 2 4 11 3 4 9	
выходные данные	Скопировать
1	
входные данные	Скопировать
3 2 1 2 1 2 3 2 1 1 2 5	
выходные данные	Скопировать
2	

Примечание

В первом примере каждое ребро полезно, кроме ребра веса 5.

Во втором примере полезно только ребро между 1 и 2, потому что оно принадлежит пути $1 - 2$, и $10 \leq 11$. С другой стороны, ребро между вершинами 3 и 4 не является полезным.

В третьем примере оба ребра полезны, например, потому что существует путь $1 - 2 - 3 - 2$ длины 5. Обратите внимание, что путь может проходить через вершины по несколько раз.

Разбор задачи F

1482F - Полезные рёбра

Посчитаем все попарные расстояния между вершинами алгоритмом Флойда за $O(n^3)$. Рассмотрим все тройки с фиксированной первой вершиной, назовём её v . Найдем все полезные рёбра только для этих троек. Ребро (a, b, w) полезно, если существует тройка (v, u_i, l_i) , такая что $dist(v, a) + w + dist(b, u_i) \leq l_i \Leftrightarrow -l_i + dist(u_i, b) \leq -w - dist(v, a)$. Заметим что справа стоит величина, зависящая только от фиксированной вершины v , и ребра. Поэтому достаточно минимизировать левую часть по всем возможным тройкам. Это можно сделать алгоритмом Дейкстры за $O(n^2)$, если изначально проинициализировать расстояние до u_i значением $-l_i$. После этого можно для каждого ребра проверить является ли оно полезным хоть для какой-то тройки. Итоговая асимптотика $O(n^3)$.

<https://codeforces.com/contest/1482/problem/G>

G. Ва-банк

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Густав — старший менеджер большого банка. У него есть неограниченный доступ в базу данных банка, и в несколько кликов он может переместить любую сумму со счета банковского резерва на свой личный счет. Однако банк недавно внедрил популярную систему, использующую искусственный интеллект, для отслеживания мошеннических операций, что затрудняет кражу.

Густав знает, что данная система просто отслеживает все операции, превышающие сумму в M евро, и такие операции проверяются вручную некоторым количеством банковских работников. Поэтому любая мошенническая операция, превышающая данный лимит, будет обнаружена, а любая меньшая операция останется незамеченной.

Густав не знает значение лимита M и хочет его найти. За одну операцию он может выбрать некоторое целое число X и попробовать переместить X евро из банковского резерва на личный счет. Затем случается следующее.

- Если $X \leq M$, то операция остается незамеченной и баланс на счете Густава увеличивается на X евро.
- В противном случае, если $X > M$, операцию обнаруживают и отменяют. Более того, в таком случае Густаву нужно оплатить штраф в X евро со своего счета. Если у него на счете меньше, чем X евро, то его увольняют и передают полиции.

Изначально на счету Густава 1 евро. Помогите ему найти точное значение M за не более чем 105 операций, не допуская увольнения.

Входные данные

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число t ($1 \leq t \leq 1000$) — количество наборов входных данных.

В каждом наборе входных данных нет никаких входных данных до вашего первого запроса. Гарантируется, что M — целое число, и $0 \leq M \leq 10^{14}$.

Выходные данные

В каждом наборе входных данных, когда вы знаете значение M , выведите одну строку формата «! M ». После этого ваша программа должна перейти к следующему набору входных данных или завершиться, если это был последний набор.

Протокол взаимодействия

Когда вы хотите сделать операцию, выведите строку формата «? X », означающую, что вы пробуете переместить X евро ($1 \leq X \leq 10^{14}$). К качестве ответа считайте одну строку, которая может принимать следующие значения:

- «Lucky!», если $X \leq M$. Баланс вашего счета увеличивается на X .
- «Fraudster!», если $X > M$. Баланс вашего счета уменьшается на X .
- «Fired!», если $X > M$, но вы не можете выплатить X евро штрафа. В таком случае ваша программа должна немедленно завершиться. Вы также получите такой ответ, если превысили максимальное число запросов.

Вы можете сделать не более чем 105 запросов такого типа в каждом наборе входных данных.

После вывода запроса не забудьте вывести перевод строки и сбросить буфер вывода. В противном случае вы получите вердикт Решение «зависло». Для сброса буфера используйте:

- `fflush(stdout)` или `cout.flush()` в C++;
- `System.out.flush()` в Java;
- `flush(output)` в Pascal;
- `stdout.flush()` в Python;
- смотрите документацию для других языков.

Взломы

Чтобы сделать взлом, подготовьте входные данные в следующем формате.

Первая строка содержит одно целое число t ($1 \leq t \leq 1000$) — количество наборов входных данных.

Каждый набор входных данных описывается строкой, содержащей одно целое число M ($0 \leq M \leq 10^{14}$).

Пример

входные данные	Скопировать
1 Lucky! Lucky! Lucky! Lucky! Lucky! Fraudster!	
выходные данные	Скопировать
? 1 ? 2 ? 3 ? 4 ? 5 ? 6 ! 5	

Примечание

В примере $M = 5$, поэтому операция $X = 6$ будет обнаружена. В этот момент баланс счета Густава составляет 16 евро, поэтому он просто платит штраф.

Разбор задачи G

1482G - Ва-банк

Разделим наше решение на две части. Для начала найдем верхнюю границу этого самого M . Для этого будет делать запросы по степеням двойки: $1, 2, 4, \dots$. Тогда нам всегда будет хватать денег на очередной запрос. Когда-то мы проиграем и получится, что ответ лежит на отрезке $[2^k, 2^{k+1}]$.

Это займет не более 47 запросов.

Теперь можно было бы сделать например такой бинпоиск: один раз взять левую границу, далее каждый раз брать левую границу и середину. Тогда если мы победили - все хорошо. Если проиграли - потеряли $L + \frac{R-L}{2}$. Заметим, что L мы уже взяли, а вот сумма $\frac{R-L}{2}$ будет не больше L , так как $R - L = L$, а длина каждый раз уменьшается вдвое.

Однако это решение требует еще $2 \log(10^{14})$ операций, что много.

Давайте заметим, что если левая часть не больше половины, то этой дополнительной L хватит, чтобы покрыть поражения. Теперь заметим, что мы никак не используем то, что после победы мы получаем M . Давайте это исправим.

Попробуем построить оптимальное решение. Пусть текущий отрезок равен (l, r) . Текущий баланс не меньше $l \cdot y + (r - l)$. Тогда, если $y = 0$ то мы должны сделать запрос в l , а если $y > 1$ - можно сделать любой запрос на отрезке.

Заметим, что после неудачного запроса наш баланс не меньше $l \cdot (y - 1) + (r - l)$ (для новых l и r), а после удачного покла предположим, что наш баланс не меньше $l \cdot (y + 1) + (r - l)$ (для новых l и r). Это не всегда так, обсудим это позже. Заметим, что при разумных действиях часть $(r - l)$ сохраняется, поэтому не будем ее рассматривать. Таким образом баланс зависит только от y .

Пусть $dp[x][y]$ такое максимальное число d , что за x запросов можно найти ответ на отрезке $(l, l + d)$ имея исходный баланс $y \cdot l + d$. В таком случае $dp[x][y] = dp[x - 1][y - 1] + dp[x - 1][y + 1]$, эти величины легко вычислить, а в решении — следовать соответствующему разбиению.

Можно показать, что $dp[k][0] = C_k^{k/2}$. Тогда можно заметить, что $k \leq 49$.

Таким образом получается 97 запросов.

Однако стоит заметить, что если мы перешли от (l, r, y) в $(m, r, y + 1)$, то на самом деле у нас не у левых границ, так как баланс был $y \cdot l$ и стал $y \cdot l + m$, а нужно $(y + 1) \cdot m$.

Можно показать, что нам хватит 3 доп запросов в исходной l .

<https://codeforces.com/contest/1482/problem/H>

Н. Экзамен

ограничение по времени на тест: 2 секунды

ограничение по памяти на тест: 512 мегабайт

ввод: стандартный ввод

вывод: стандартный вывод

В этом году в Конохе вновь проводится экзамен на звание Чунина, и в нем участвуют n ниндзя с именами s_1, s_2, \dots, s_n . Все имена участников различны. Один из этапов экзамена — сражения между участниками. В этом году руководство Конохи решило, что определять, кто с кем сражается, будут по-новому. Ниндзя i сражается с ниндзя j , если выполнены три правила:

- $i \neq j$;
- s_j — подстрока s_i ;
- не существует k , отличного от i и j такого, что s_j — подстрока s_k , а s_k — подстрока s_i .

Строка a является подстрокой b , если a может быть получена из b удалением нескольких (возможно, ни одного или всех) символов из начала и нескольких (возможно, ни одного или всех) символов из конца.

Необходимо выяснить, сколько сражений будет проведено на экзамене в этом году.

Входные данные

Первая строка содержит натуральное число n ($1 \leq n \leq 10^6$) — число участников экзамена.

Следующие n строк содержат имена участников. Все имена непустые, различны и состоят из строчных латинских букв. Суммарная длина всех имен не превосходит 10^6 .

Выходные данные

Необходимо вывести единственное число — количество сражений на экзамене в этом году.

Примеры

входные данные	Скопировать
5 hidan dan hanabi bi nabi	
выходные данные	Скопировать
3	
входные данные	Скопировать
4 abacaba abaca acaba aca	
выходные данные	Скопировать
4	

Примечание

В первом примере `hidan` сражается с `dan`, а `hanabi` сражается с `nabi`, который также сражается с `bi`. Ниндзя с именами `hanabi` и `bi` не сражаются, поскольку есть ниндзя с именем `nabi`, из-за которого для этой пары не выполняется третье условие.

Во втором примере сражения происходят между `abacaba` и `acaba`, `abacaba` и `abaca`, `acaba` и `aca`, `abaca` и `aca`.

Разбор задачи N

1482N - Экзамен

Зафиксируем конкретную строку и найдём все рёбра выходящие из этой строки. Для каждой позиции i нашей строки найдём значение $left_i$ — позиция, в которой начинается наидлиннейшая подстрока, равная одной из строк нашего словаря, и заканчивающаяся в позиции i . Легко понять, что рёбра могут идти только в такие строки, и никакие другие. Осталось понять какие из этих строк лишние.

Строка является лишней, когда у неё есть вхождение в нашу строку, в котором она накрывается вхождением какой-либо более длинной строки. Пусть $ind_i = \min\{left_j, j > i\}$. Тогда лишними будут все строки из нашего словаря, которые являются суффиксами подстроки $[ind_i, i]$, для какого-либо i . Их можно находить с помощью структуры Ахо-Корасик. Так как суффиксные ссылки в Ахо-Корасик образуют дерево, то достаточно найти вершины соответствующие подстрокам $[ind_i, i]$, и покрасить путь по суффиксным ссылкам до корня дерева. После этого можно просто проверить покрашены ли вершины, соответствующие подстроками $[left_i, i]$. Чтобы не красить путь наивным образом, можно воспользоваться структурой Деревя Отрезков или `std::set`. Итоговая асимптотика $O(n \cdot \log n)$.