Задача A. Объединение

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

В конгломерат входит n компаний. Для упрощения управления владельцы решили объединить все компании в одну. По закону разрешается объединять только две компании, поэтому владельцы планируют по очереди выбирать две компании и объединять их в одну, а затем повторять процесс, пока не останется только одна компания.

При этом антимонопольная служба запрещает объединять компании, если она подозревает, что происходит недружественное поглощение. В качестве критерия используется сравнение максимальных зарплат в объединяемых компаниях. Две компании можно объединить в одну, только если максимальные зарплаты в них совпадают.

Чтобы выполнить требование антимонопольной службы, владельцы могут перед объединением изменить зарплаты в одной из объединяемых компаний. Однако профсоюз настаивает на двух требованиях: разрешается только увеличивать зарплаты, кроме того, всем сотрудникам одной и той же компании необходимо увеличивать зарплату на одну и ту же величину.

Разумеется, руководство хочет минимизировать суммарное повышение всех зарплат. Помогите им выяснить, на какую минимальную суммарную величину придется повысить зарплаты сотрудников всех компаний в процессе объединения их в одну компанию.

Формат входных данных

В первой строке входных данных находится целое число n — количество компаний в конгломерате ($1 \leqslant n \leqslant 2 \cdot 10^5$). В следующих n строках описаны сами компании, каждая в своей строке.

Описание каждой компании начинается с целого числа m_i — количества сотрудников в ней ($1 \leqslant m_i \leqslant 2 \cdot 10^5$). Далее идут m_i целых чисел — зарплаты сотрудников. Все зарплаты положительны и не превосходят 10^9. Гарантируется, что общее количество сотрудников во всех компаниях не превосходит $2 \cdot 10^5$.

Формат выходных данных

Выведите единственное число — минимальную суммарную величину, на которую придется повысить зарплаты сотрудникам.

Пример

стандартный ввод	стандартный вывод
3 2 4 3 2 2 1 3 1 1 1	13

Замечание

Один из оптимальных вариантов объединения компаний в примере следующий. Сначала объединим первую и вторую компании, перед этим повысив зарплаты во второй компании на 2. Теперь в конгломерате две компании с зарплатами $[4, 3, 4, 3]$ и $[1, 1, 1]$. Чтобы их объединить, повысим зарплаты во второй из них на 3. Итого суммарное увеличение составило $2 + 2 + 3 + 3 + 3 = 13$.

Задача B. Эксперт LaTeX

Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Девочка Регина заканчивает написание дипломной работы. С версткой она попросила помочь мальчика Гришу, который является экспертом LaTeX.

Гриша с легкостью справился с версткой работы, однако столкнулся с проблемой при оформлении списка использованных источников. По требованиям к оформлению дипломной работы, источники должны быть перечислены в списке в том же порядке, в каком они встречались в тексте работы. Ссылка на каждый источник в работе Регины встречается ровно один раз.

Для того, чтобы вставить ссылку на источник, используется конструкция `\cite{<reference>}`, где `<reference>` — название ссылки на источник. В листинге ниже представлен текст с тремя ссылками на источники.

```
The most famous characters of Pushkin's works are Onegin \cite{onegin},
Dubrovsky \cite{dubrovsky} and Tsar Saltan \cite{saltan}.
```

Сам список использованных источников пишется после текста и задается следующей конструкцией:

```
\begin{thebibliography}{99}
\bibitem{onegin} A.S.Pushkin. Eugene Onegin. 1831.
\bibitem{dubrovsky} A.S.Pushkin. Dubrovsky. 1841.
\bibitem{saltan} A.S.Pushkin. The Tale of Tsar Saltan. 1832.
\end{thebibliography}
```

Здесь 99 обозначает максимальное число источников, а после `\bibitem{<reference>}` идет какое-либо описание источника.

Текст оказался довольно объемный, а Гриша довольно ленивый, и ему лень проверять упорядоченность источников в списке. Поэтому он просит вас написать программу, которая по тексту и списку использованных источников определяет, корректно он упорядочен или нет, и если нет, то строит корректный список в нужном формате.

Формат входных данных

Во входных данных представлен текст работы, который состоит из строчных и заглавных латинских букв, переводов строк, пробелов, символов «.», «,», «'» (ASCII код 39) и конструкций `\cite{<reference>}`. Здесь `<reference>` представляет собой непустую строку, состоящую из не более чем 100 строчных латинских символов. ASCII код символа «\» — 92.

Каждая конструкция `\cite{<reference>}` отделена от предшествующего текста пробелом или переводом строки. В тексте могут быть пустые строки.

В следующей строке после текста содержится строка `\begin{thebibliography}{99}`, после которой в каждой следующей строке задано описание источника в описанном выше формате. Описание источника состоит из тех же символов, которые допустимы в тексте. Длина описания каждого источника не превосходит 100 символов. После списка идет строка `\end{thebibliography}`.

Гарантируется:

- Суммарное число источников не превосходит 99.
- В тексте встречается хотя бы один источник.
- Если ссылка на источник есть в тексте, то этот источник есть и в списке источников.
- Число конструкций `\cite{<reference>}` в тексте совпадает с числом конструкций `\bibitem{<reference>}` в списке.
- Ссылки на источники в тексте различны между собой.
- Суммарное число строк во вводе не превосходит 10^5.
- Сумма длин всех строк в тексте не превосходит 10^5.
- Во вводе не встречается двух подряд пробелов, первый и последний символ каждой строки — не пробелы.

Формат выходных данных

Выведите `Correct`, если список использованных источников составлен верно. Иначе выведите `Incorrect`. Затем, начиная со второй строки, выведите корректный список источников в нужном формате.

Описание источников должно совпадать с заданным во вводе.

Примеры

стандартный ввод
The most famous characters of Pushkin's works are Onegin \cite{onegin}, Dubrovsky \cite{dubrovsky} and Tsar Saltan \cite{saltan}. \begin{thebibliography}{99} \bibitem{saltan} A.S.Pushkin. The Tale of Tsar Saltan. 1832. \bibitem{onegin} A.S.Pushkin. Eugene Onegin. 1831. \bibitem{dubrovsky} A.S.Pushkin. Dubrovsky. 1841. \end{thebibliography}
стандартный вывод
Incorrect \begin{thebibliography}{99} \bibitem{onegin} A.S.Pushkin. Eugene Onegin. 1831. \bibitem{dubrovsky} A.S.Pushkin. Dubrovsky. 1841. \bibitem{saltan} A.S.Pushkin. The Tale of Tsar Saltan. 1832. \end{thebibliography}

стандартный ввод
The most famous characters of Pushkin's works are Onegin \cite{onegin}, Dubrovsky \cite{dubrovsky} and Tsar Saltan \cite{saltan}. \begin{thebibliography}{99} \bibitem{onegin} A.S.Pushkin. Eugene Onegin. 1831. \bibitem{dubrovsky} A.S.Pushkin. Dubrovsky. 1841. \bibitem{saltan} A.S.Pushkin. The Tale of Tsar Saltan. 1832. \end{thebibliography}
стандартный вывод
Correct

Задача C. Новогодние подарки

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Санта-Клаус приготовил коробки с подарками для n детей, по одной коробке каждому ребёнку. В коробках есть m видов подарков: воздушные шары, конфеты, шоколадки, игрушечные машинки... Любой ребёнок расстроится, получив два одинаковых подарка, поэтому все подарки в каждой коробке различны.

Только упаковав все коробки, Санта заметил, что в разных коробках разное число подарков. Это несправедливо по отношению к детям! Санта решил переместить некоторые подарки из одной коробки в другую, чтобы в итоге размеры самой большой и самой маленькой коробки отличались как можно меньше. Все подарки в каждой коробке по-прежнему должны оказаться различными. Санта хочет расправиться с работой побыстрее, поэтому ему нужно минимизировать число перемещений.

Вам даны списки подарков в каждой коробке. Найдите кратчайшую последовательность перемещений подарков между коробками, в результате которой размеры самой большой и самой маленькой коробки будут отличаться как можно меньше, и все подарки в каждой коробке будут различны.

Формат входных данных

В первой строке содержатся два целых числа n и m ($1 \leqslant n, m \leqslant 100\,000$) — количество коробок и видов подарков. Будем обозначать виды подарков целыми числами от 1 до m.

Каждая из следующих n строк содержит описание коробки. Описание начинается с целого числа s_i ($s_i \geqslant 0$) — количества подарков в коробке. Затем идут s_i различных чисел между 1 и m — виды подарков в данной коробке.

Суммарное количество подарков во всех коробках не превосходит 500 000.

Формат выходных данных

В первой строке выведите одно число k — минимальное число перемещений подарков между коробками, которое позволяет Санте добиться желаемого. Затем выведите k строк с описаниями перемещений. Перемещения должны быть выведены в том же порядке, в котором их требуется применить. Каждое перемещение описывается тремя числами $from_i$, to_i, $kind_i$. Это значит, что подарок вида $kind_i$ перемещается из коробки с номером $from_i$ в коробку с номером to_i. Коробки нумеруются с единицы в порядке следования во входных данных.

Перед перемещением хотя бы один подарок вида $kind_i$ должен находиться в коробке с номером $from_i$. После окончания всех перемещений ни в какой коробке не должно находиться двух подарков одного вида.

Если оптимальных последовательностей перемещений несколько, выведите любую из них.

Пример

стандартный ввод	стандартный вывод
3 5 5 1 2 3 4 5 2 1 2 2 3 4	2 1 3 5 1 2 3

Задача D. Похожие массивы

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

У Васи был массив из n натуральных чисел от 1 до n. Он выбрал m различных пар различных позиций и записал их на листочек. Затем Вася для каждой пары позиций сравнил между собой находящиеся на них элементы и записал на другой листочек результаты сравнений — «больше», «меньше» или «равно».

Через несколько лет он нашёл свой первый листочек, а второй найти не смог. Кроме того, он совершенно забыл, какой у него был исходный массив. В частности, Вася забыл, были ли в его массиве равные элементы. Он рассказал эту трагическую историю своей учительнице информатики Елене Андреевне.

В ответ она сказала, что может так случиться, что даже если Вася найдет свой второй листочек, информации, записанной на нём, может оказаться недостаточно, чтобы определить, были ли в массиве два равных элемента.

Вася тут же захотел найти два массива натуральных чисел длины n, все элементы первого массива должны быть различны, а во втором массиве должно быть хотя бы два равных элемента. При этом для каждой пары индексов, записанных на первом листочке, результаты сравнений в обоих массивах будут одинаковы.

Помогите Васе найти два массива длины n, удовлетворяющие данным условиям, или выясните, что для его множества пар индексов таких массивов не существует.

Формат входных данных

В первой строке задано два целых числа n, m — количество чисел в массиве и количество сравнений, сделанных Васей ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$).

Каждая из следующих m строк содержит два целых числа a_i, b_i — номера позиций пары i-го сравнения ($1 \leqslant a_i, b_i \leqslant n$; $a_i \neq b_i$), гарантируется, что каждая неупорядоченная пара встречается во вводе не более одного раза.

Формат выходных данных

В первой строке выведите «YES», если существуют два массива, для которых результаты сравнений будут одинаковы, все числа первого из которых различны, а второй содержит хотя бы одну пару одинаковых чисел. Иначе выведите «NO».

Если такие массивы существуют, то во второй строке выведите массив из различных чисел, а в третьей строке — массив, содержащий хотя бы одну пару равных чисел, соответствующие условию. Элементы массивов должны быть целыми числами от 1 до n.

Примеры

стандартный ввод	стандартный вывод
1 0	NO
3 1 1 2	YES 1 3 2 1 3 1
4 3 1 2 1 3 2 4	YES 1 3 4 2 1 3 4 1

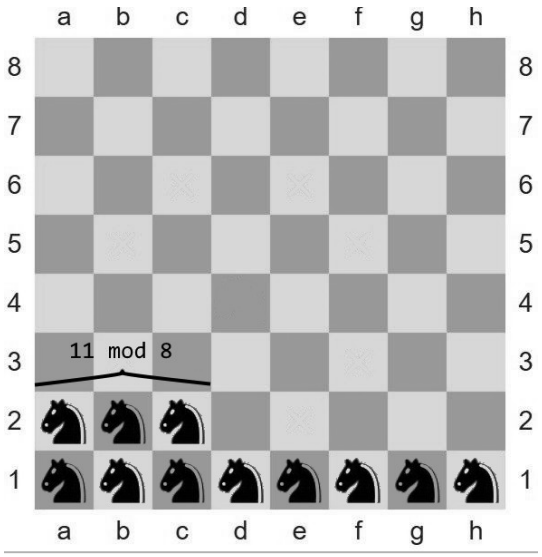
Задача E. Конный спорт

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Как известно, шахматы — это вид спорта. Однако далеко не все с этим согласны. Например, первокурсник Дима, занимающийся конным спортом, считает, что такую скучную вещь как шахматы спортом назвать никак нельзя. Когда его друг-шахматист Саша об этом узнал, он сразу решил, что надо показать Диме, насколько сложными и интересными на самом деле бывают шахматы, и дал Диме задачку, решать которую тому бы точно понравилась.

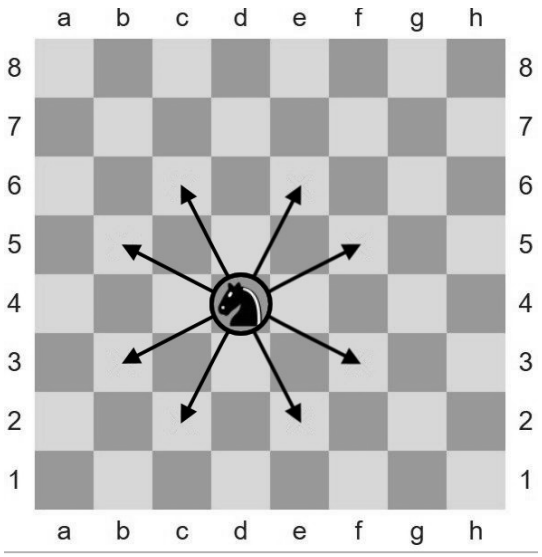
На шахматной доске 8×8 Саша расставил k коней. Эти кони давно вышли на прогулку и хотят поскорее вернуться в стойла. К сожалению, кони не помнят пути обратно.

Будем говорить, что кони находятся в стойлах, если выполнены следующие условия: несколько (а именно, k div 8) нижних строк доски полностью заполнены конями, а следующая строка может содержать оставшихся коней в нескольких самых левых клетках (если $k \bmod 8 \neq 0$, то $k \bmod 8$ коней занимают самые левые клетки следующей строки).



В этих клетках, по одному в каждой, в конце должны оказаться $k = 11$ коней.

Разумеется, поскольку это шахматная задача, то все кони тоже ходят как шахматные — ровно на две клетки по одной координате и ровно на одну по другой.



На данном рисунке показаны возможные ходы шахматного коня.

Кони делают ходы по одному. В каждый момент времени в одной клетке может находиться не более одного коня.

За два дня решения этой задачи Дима понял, что шахматы не такие уж и скучные. Но всё же она ему немного надоела, так что он просит вас о помощи — найдите такой порядок ходов коней, чтобы в каждый момент времени в каждой клетке было не более одного коня, и в конце кони находились в стойлах. Минимизировать количество ходов не требуется, но необходимо сделать не слишком много перемещений коней — не больше 1500.

Формат входных данных

В первой строке входных данных задано число k — количество коней на доске ($1 \leqslant k \leqslant 64$). Далее следуют k строк с описаниями коней — на каждой строке записана позиция коня на поле в формате `xy`, где `x` — буква столбца, а `y` — номер строки.

Столбцы обозначены буквами от «`a`» до «`h`» слева направо, строки пронумерованы цифрами от 1 до 8 снизу вверх.

Формат выходных данных

В первой строке выведите суммарное количество ходов, за которое кони добираются до стойл. Затем выведите по одному в строке ходы коней в порядке их совершения, каждый в формате `xy-zt`, где `x` и `z` — столбцы, а `y` и `t` — строки.

Обратите внимание, не требуется минимизировать число ходов, но их число должно быть не более 1500.

Примеры

стандартный ввод	стандартный вывод
2 a5 f1	4 a5-b3 b3-a1 f1-d2 d2-b1
3 a1 b3 c2	5 b3-d2 d2-b1 c2-b4 b4-d3 d3-c1

Задача F. Как узнать свои баллы

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Это интерактивная задача.

Петя участвует в контесте, где участники узнают результаты тестирования каждой из своих посылок только после конца соревнования.

В контесте, в котором участвовал Петя, n задач, пронумерованных от 1 до n. По каждой из этих задач он сдал решение. После соревнования он очень волновался — ведь от того, сколько баллов он получил по каждой задаче, зависит то, какое место он занял. К сожалению, результаты станут известны только на награждении, до которого еще несколько часов.

Петя решил заранее узнать свой результат — выпытать у жюри количество баллов, полученное им по каждой задаче, задавая им странные вопросы. Пусть Петя получил $c_1, c_2, \ldots, c_n$ баллов по задачам $1, 2, \ldots, n$, соответственно. Петя не знает эти числа, но хочет их узнать.

Петя называет номера трех различных задач контеста и просит жюри сказать ему сумму минимального и максимального количества баллов, полученных им по этим трем задачам. К счастью, жюри, ничего не подозревая, отвечает на его странные вопросы. Более формально, если Петя спросит про три задачи i, j, k, то в ответ ему скажут число $min(c_i, c_j, c_k) + max(c_i, c_j, c_k)$.

Помогите Пете отгадать числа $c_1, c_2, \ldots, c_n$, задав не более $4n$ вопросов членам жюри.

Протокол взаимодействия

Сначала на вход вашей программе подается одно целое число n — количество задач, которое было в контесте ($5 \leqslant n \leqslant 1000$).

После этого ваша программа может делать запросы: чему равна сумма минимального и максимального баллов, полученных Петей по задачам i, j, k. Для этого нужно вывести в выходной поток в отдельной строке «? i j k» ($1 \leqslant i, j, k \leqslant n$, $i \neq j$, $j \neq k$, $i \neq k$). В ответ на запрос во входном потоке будет записано одно целое число $min(c_i, c_j, c_k) + max(c_i, c_j, c_k)$.

Если ваша программа сделает более $4n$ запросов или задаст некорректный запрос, она получит вердикт «`Wrong Answer`».

Определив баллы Пети, ваша программа должна вывести в выходной поток «! c_1 c_2 ... c_n» — баллы, полученные Петей по каждой из задач.

Гарантируется, что баллы, полученные Петей по каждой из задач, являются целыми числами и удовлетворяют ограничениям $0 \leqslant c_i \leqslant 10^9$ для всех $1 \leqslant i \leqslant n$. Также все числа c_i фиксированы заранее и не будут меняться в ходе тестирования вашей программы.

Пример

стандартный ввод	стандартный вывод
5 2 4 4 1	? 1 2 3 ? 1 3 5 ? 5 4 3 ? 1 4 2 ! 1 0 2 1 3

Замечание

После каждого действия вашей программы выводите символ перевода строки. Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java или «`print`» в Python, сброс потока вывода у вас происходит автоматически, дополнительно делать «`flush`» не обязательно. Если вы используете другой способ вывода, рекомендуется делать «`flush`», но все равно обязательно требуется выводить символ перевода строки.

Пояснение к примеру

В тесте из условия было 5 задач в контесте, и Петя получил 1, 0, 2, 1, 3 балла по соответствующим задачам. Если он спросит про задачи 1, 2, 3, то минимальное количество баллов, полученное им по этим трем задачам равно 0 (по 2-й задаче), а максимальное равно 2 (по 3-й задаче). Таким образом, жюри скажет ему число $2 = 0 + 2$ в ответ на этот вопрос.

Задача G. Комбокамень

Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Одна большая компания разрабатывает компьютерную игру «Комбокамень», которая должна мигом перевернуть всю индустрию. Правила игры достаточно сложные, и на реализацию серверного движка, моделирующего ход игры, был объявлен конкурс. От вас требуется реализовать подобный серверный движок.

Суть игры — игроки умеют призывать на арену и усиливать существ, используя заклинания, а также заставлять их сражаться друг с другом. Каждое существо имеет два параметра — численное значение атаки a и численное значение оставшегося здоровья h. Для краткости будем обозначать параметры существа как (a, h). Исходно на арене нет существ.

Игроку доступны следующие заклинания:

- *Призыв существа*: Призвать новое существо с характеристиками $(1, 1)$. Если уже в игру было введено k существ, то новое существо получает номер $k + 1$.
- *Благословение силы*: Удвоить атаку выбранного существа. Если до применения этого заклинания оно имело характеристики (a, h), то после этого действия оно будет иметь характеристики $(2a, h)$.
- *Божественный дух*: Удвоить здоровье выбранного существа. Если до применения этого заклинания оно имело характеристики (a, h), то после этого действия оно будет иметь характеристики $(a, 2h)$.
- *Копия из лавы*: Призвать новое существо, которое будет иметь такие же характеристики, как и выбранное заклинанием существо. Если уже в игру было введено k существ, то новое существо получает номер $k + 1$.
- *Сражайся!*: Заставить двух различных существ сразиться. Во время сражения оба существа одновременно наносят друг другу по одному удару, уменьшая количество здоровья соперника на значение своей атаки. Так, если сражаются два существа с характеристиками (a_1, h_1) и (a_2, h_2), то после сражения они будут иметь характеристики $(a_1, h_1 - a_2)$ и $(a_2, h_2 - a_1)$, соответственно. Если после сражения у существа остается 0 или меньше единиц здоровья, оно умирает и больше не может участвовать в игре.

От серверного движка, на реализацию которого объявлен конкурс, требуется способность промоделировать все события и для каждого созданного во время игры существа вывести номер хода, на котором оно погибло, либо определить, что оно осталось живо к концу игры.

Кроме того, движок должен корректно обрабатывать случаи, когда игрок пытается взаимодействовать с мертвыми по мнению сервера существами: если заклинание *Благословение силы*, *Божественный дух* или *Сражайся!* обращено к уже мертвому существу, то не должно произойти ничего. Если заклинание *Копия из лавы* применено к мертвому существу, создается его мертвая копия с такими же характеристиками, но умершая на текущем ходу, в момент копирования.

Формат входных данных

В первой строке дано число n — количество совершенных ходов ($1 \leqslant n \leqslant 250\,000$).

В следующих n строках даны ходы, пришедшие к серверному движку, в следующем формате:

- 1 — применить заклинание *Призыв существа*;
- 2 i — применить заклинание *Благословение силы* к существу с номером i;
- 3 i — применить заклинание *Божественный дух* к существу с номером i;
- 4 i — применить заклинание *Копия из лавы* к существу с номером i;
- 5 i j — применить заклинание *Сражайся!* к существам с номерами i и j.

Гарантируется, что любые упомянутые в запросах существа к моменту запроса уже были призваны, но, возможно, могут уже быть мертвы.

Формат выходных данных

В первой строке выведите одно целое число k — количество существ, призванных за время игры.

В следующей строке выведите k целых чисел $t_1, t_2, \ldots, t_k$ — если существо с номером i осталось живо к концу игры, то t_i должно быть равно -1, иначе t_i должно быть равно номеру хода, на котором оно погибло.

Пример

стандартный ввод	стандартный вывод
16 1 2 1 3 1 1 5 1 2 3 1 1 3 3 3 3 4 1 5 1 3 3 3 5 1 3 5 4 3 5 4 3 4 1	5 13 5 14 -1 16

Замечание

В таблице можно увидеть, как изменялись характеристики существ в первом примере.

ход	1	2	3	4	5
0	-	-	-	-	-
1	(1, 1)	-	-	-	-
2	(2, 1)	-	-	-	-
3	(2, 2)	-	-	-	-
4	(2, 2)	(1, 1)	-	-	-
5	(2, 1)	мертво	-	-	-
6	(2, 2)	мертво	-	-	-
7	(2, 2)	мертво	(1, 1)	-	-
8	(2, 2)	мертво	(1, 2)	-	-
9	(2, 2)	мертво	(1, 4)	-	-
10	(2, 2)	мертво	(1, 4)	(2, 2)	-
11	(2, 1)	мертво	(1, 2)	(2, 2)	-
12	(2, 1)	мертво	(1, 4)	(2, 2)	-
13	мертво	мертво	(1, 2)	(2, 2)	-
14	мертво	мертво	мертво	(2, 1)	-
15	мертво	мертво	мертво	(2, 1)	-
16	мертво	мертво	мертво	(2, 1)	мертво

Задача N. Линеаризация

Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Побитовое «и» (and) двух целых неотрицательных чисел устроено следующим образом: запишем оба числа в двоичной системе счисления, i-й двоичный разряд результата равен 1, если у обоих аргументов он равен 1. Например, $(14 \text{ and } 7) = (1110_2 \text{ and } 0111_2) = 110_2 = 6$.

«Исключающее или» (xor) двух двоичных цифр равно 1, если они различны, и 0, если они равны. Таким образом, $0 \text{ xor } 0 = 0$, $0 \text{ xor } 1 = 1$, $1 \text{ xor } 0 = 1$ и $1 \text{ xor } 1 = 0$.

Функция четности $P(x)$ для целого неотрицательного числа x равна 1, если в двоичной записи числа x нечетное число единичных разрядов, либо 0, если в двоичной записи x четное число единичных разрядов. Например, $P(5) = P(101_2) = 0$, $P(7) = P(111_2) = 1$.

Рассмотрим строку из нулей и единиц, длина которой является степенью двойки: $s = s_0 s_1 \dots s_{n-1}$, где $n = 2^k$. Будем называть строку линейной, если найдется такое число x, $0 \leqslant x < n$, и такая двоичная цифра b, что для всех i от 0 до $n-1$ выполнено $s_i = P(i \text{ and } x) \text{ xor } b$.

Например, строка «1100» является линейной: выберем $x = 2 = 10_2$ и $b = 1$.

- $s_0 = P(0 \text{ and } 2) \text{ xor } 1 = P(0) \text{ xor } 1 = 0 \text{ xor } 1 = 1$;
- $s_1 = P(1 \text{ and } 2) \text{ xor } 1 = P(0) \text{ xor } 1 = 0 \text{ xor } 1 = 1$;
- $s_2 = P(2 \text{ and } 2) \text{ xor } 1 = P(2) \text{ xor } 1 = 1 \text{ xor } 1 = 0$;
- $s_3 = P(3 \text{ and } 2) \text{ xor } 1 = P(2) \text{ xor } 1 = 1 \text{ xor } 1 = 0$.

Строка же «0001» не является линейной: какое бы значение x мы ни выбрали, выполнено равенство $P(0 \text{ and } x) = P(0) = 0$, значит $b = 0$. Но $0 = P(1 \text{ and } x)$ и $0 = P(2 \text{ and } x)$, следовательно $x = 0$. Однако $P(3 \text{ and } 0) = 0 \neq s_3 = 1$.

Пусть задана строка из нулей и единиц. За одно действие разрешается взять отрезок подряд идущих символов в строке и заменить их на противоположные: нули на единицы, а единицы на нули. Назовём *трудностью линеаризации* строки минимальное количество действий, за которое её можно сделать линейной.

Трудность линеаризации строки «0001», например, равна 1: можно инвертировать отрезок с 0-го по 2-й символ, получится строка «1111», которая является линейной с $x = 0$, $b = 1$. Есть и другие способы сделать эту строку линейной за 1 действие.

Задана строка t и q запросов (l_i, r_i). Для каждого запроса рассмотрим в качестве строки s подстроку строки t с l_i-го по r_i-й символ, включительно. Символы строки t пронумерованы от начала, начиная с 0. Гарантируется, что длина каждой строки-запроса является степенью двойки. Требуется выяснить трудность линеаризации каждой такой подстроки.

Формат входных данных

Первая строка входных данных содержит число m — длину строки t ($1 \leqslant m \leqslant 200\,000$). Вторая строка содержит строку t, она имеет длину m и состоит из нулей и единиц.

Следующая строка содержит число q — количество запросов ($1 \leqslant q \leqslant 200\,000$). Каждая из следующих q строк содержит по два целых числа: l_i и r_i ($0 \leqslant l_i \leqslant r_i < m$, $r_i - l_i + 1 \geqslant 2$ и является степенью двойки).

Формат выходных данных

Для каждого запроса выведите одно число: трудность линеаризации соответствующей подстроки строки t.

Пример

стандартный ввод	стандартный вывод
8 00000101 3 0 7 2 5 0 3	2 1 0

Замечание

В примере в первом запросе требуется линеаризовать всю строку. Это можно сделать, например, инвертировав отрезок с 4-го по 6-й символ, получив строку «00001011», и затем инвертировав 5-й символ, получив строку «00001111». Она является линейной с $x = 4$ и $b = 0$.

Во втором запросе необходимо линеаризовать строку «0001», это можно сделать за 1 действие, как описано в условии.

В третьем запросе необходимо линеаризовать строку «0000», которая и так является линейной для $x = 0$, $b = 0$.

Задача I. Минимальное произведение

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Задан массив целых чисел $a_1, \ldots, a_n$. Требуется найти такие i и j, что $i < j$, $a_i < a_j$, и произведение $a_i \cdot a_j$ минимальное возможное.

Формат входных данных

Входные данные содержат один или несколько тестов. В первой строке задано целое число t — количество тестов ($1 \leqslant t \leqslant 10^4$). Каждая из следующих t строк задаёт один тест.

Каждый тест генерируется по следующему алгоритму. Тест задаётся целыми числами n, l, r, x, y, z, b_1, b_2 ($2 \leqslant n \leqslant 10^7$, $-2 \cdot 10^9 \leqslant l \leqslant r \leqslant 2 \cdot 10^9$, $0 \leqslant x, y, z, b_1, b_2 < 2^{32}$). Число n задаёт длину массива.

Сначала генерируется последовательность b_i длины n, b_1 и b_2 заданы, а для $i > 2$ $b_i = (b_{i-2}x + b_{i-1}y + z) \bmod 2^{32}$. Затем для всех i от 1 до n присваивается $a_i = (b_i \bmod (r - l + 1)) + l$ (таким образом, $-2 \cdot 10^9 \leqslant a_i \leqslant 2 \cdot 10^9$).

Обратите внимание, что при генерации последовательности следует опасаться переполнения целочисленных типов, рекомендуется использовать 64-битный тип данных.

Сумма n по всем тестам не превосходит $2 \cdot 10^7$.

Формат выходных данных

Для каждого теста выведите в отдельной строке минимально возможное значение $a_i \cdot a_j$. Если не существует таких i и j, что $i < j$ и $a_i < a_j$, выведите «IMPOSSIBLE».

Пример

стандартный ввод	стандартный вывод
2 4 -5 5 11 13 17 0 3 5 0 100 0 1 0 42 42	-15 IMPOSSIBLE

Замечание

Рассмотрим подробнее генерацию массива в первом тесте.

Сначала генерируется массив b.

- $b_1 = 0$;
- $b_2 = 3$;
- $b_3 = (11 \cdot 0 + 13 \cdot 3 + 17) \bmod 2^{32} = 56$;
- $b_4 = (11 \cdot 3 + 13 \cdot 56 + 17) \bmod 2^{32} = 778$.

Теперь по нему генерируется массив a.

- $a_1 = (0 \bmod (5 - (-5) + 1)) + (-5) = (0 \bmod 11) - 5 = -5$;
- $a_2 = (3 \bmod 11) - 5 = -2$;
- $a_3 = (56 \bmod 11) - 5 = -4$;
- $a_4 = (778 \bmod 11) - 5 = 3$.

Таким образом, заданный массив $a = [-5, -2, -4, 3]$. Ответ равен $-5 \cdot 3 = -15$.

Во втором тесте массив имеет вид $[42, 42, 42, 42, 42]$.

Задача J. Два префикса

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Сегодня Миша в очередной раз пришел на урок математики, не сделав домашнее задание. Чтобы наказать ученика, учитель Андрей Владимирович решил дать ему очень сложное, но очень бессмысленное задание.

Андрей Владимирович написал на доске две строки s и t, состоящие из строчных латинских букв. Он напомнил ученикам, что *префиксом* строки называется строка, полученная удалением нескольких (возможно, нуля) последних символов из исходной строки, а *конкатенацией* двух строк называется строка, полученная приписыванием второй строки в конец к первой.

После этого учитель попросил выписать на доску все строки, являющиеся конкатенацией некоторого непустого префикса строки s и некоторого непустого префикса строки t. Когда Миша сделал это, Андрей Владимирович спросил, сколько различных строк есть среди выписанных им. Миша потратил практически весь урок, выполняя это задание, но справился с ним. Чтобы больше не тратить время на это, он попросил вас написать программу, которая выполнит это задание сама.

Формат входных данных

Первая строка входных данных содержит строку s, состоящую из строчных латинских букв. Вторая строка содержит строку t, состоящую из строчных латинских букв.

Длины каждой из строк не превосходят 10^5.

Формат выходных данных

Выведите единственное число — количество различных строк, которые являются конкатенацией некоторого непустого префикса s и некоторого непустого префикса t.

Примеры

стандартный ввод	стандартный вывод
aba aa	5
aaaaa aaaa	8

Замечание

Рассмотрим первый тест из примера. У строки s есть три непустых префикса: {a, ab, aba}. У строки t есть два непустых префикса: {a, aa}. Всего на доску будет выписано пять различных строк: {aa, aaa, aba, abaa, abaaa}. При этом строка abaa будет выписана два раза.

Во втором тесте из примера на доску будет выписано восемь различных строк: {aa, aaa, aaaa, aaaaa, aaaaaa, aaaaaaa, aaaaaaaa, aaaaaaaaa}.

Задача K. Расширение сознания вправо

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Как-то раз n человек решили расширить своё сознание.

Изначально сознание i-го человека — это две строки s_i и t_i, состоящие из маленьких латинских букв. После расширения сознание человека становится бесконечной вправо строкой $w_i = s_i + t_i + t_i + \cdots$. То есть, w_i — это конкатенация s_i и бесконечного количества t_i. Например, если $s_i =$ `mi`, $t_i =$ `nd`, то $w_i =$ `mindndndnd`$\ldots$.

Два человека с расширенным сознанием *интересны друг другу*, если сознание первого человека является подпоследовательностью сознания второго, а сознание второго является подпоследовательностью сознания первого. Бесконечная строка a называется подпоследовательностью бесконечной строки b, если существует такая бесконечная последовательность индексов $1 \leqslant i_1 < i_2 < i_3 < \ldots$, что для любого j выполнено $a_j = b_{i_j}$. Например, «`baaa`$\ldots$» — подпоследовательность «`cababababab`$\ldots$».

После расширения сознания люди решили разбиться на группы таким образом, чтобы в каждой группе любые два человека были интересны друг другу. Ваша задача — разбить их на группы так, чтобы получилось минимально возможное число групп.

Формат входных данных

Первая строка содержит одно целое число n ($1 \leqslant n \leqslant 100\,000$) — количество людей, которые решили расширить своё сознание.

Каждая из следующих n строк содержит две непустые строки s_i и t_i, состоящие только из строчных латинских букв — сознания людей до расширения.

Гарантируется, что суммарная длина всех строк не превосходит $1\,000\,000$.

Формат выходных данных

В первой строке выведите одно целое число — минимальное количество групп в разбиении.

Затем выведите каждую группу следующим образом: сначала выведите количество людей в этой группе, а затем индексы этих людей во входных данных.

Каждый из индексов нужно вывести ровно один раз. Группы и индексы внутри группы можно выводить в любом порядке. Если существует несколько оптимальных разбиений, то можно вывести любое из них.

Примеры

стандартный ввод	стандартный вывод
5 ab ab ababab ab a abb x y z w	3 3 1 2 3 1 4 1 5
3 kokoko tlin koko kotlin ko kokotlin	2 1 1 2 2 3

Замечание

В первом примере сознания людей после расширения сознания выглядят следующим образом:

- «abababab...»;
- «abababab...»;
- «aabbabbabb...»;
- «xyyyyyy...»;
- «zwwwwww...».

Ни человек 4, ни 5 не будут интересны кому-нибудь ещё. Но первые три человека попарно интересны друг другу, а значит можно составить разбиение на группы $\{1, 2, 3\}$, $\{4\}$, $\{5\}$.

Задача L. Университет Берляндии

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

В лучшем университете Берляндии учатся t студентов. В Берляндии изучают только программирование, поэтому в университете есть только один предмет. Каждый студент должен посещать лекции по программированию.

Весь курс обучения состоит из n лекций. Известно, что если студент посетит хотя бы k лекций, он успешно освоит курс программирования.

В университете есть только две аудитории: первая вмещает в себя a человек, а вторая — b человек. Чтобы всем было удобнее, руководство университета решило, что лекции с нечётными номерами будут проводиться в первой аудитории, а с чётными номерами — во второй. Так что первая лекция будет проводиться в первой аудитории, вторая лекция будет проводиться во второй аудитории, третья лекция — в первой аудитории и так далее.

Студенты поняли, что размеры аудитории могут быть настолько малы, что, возможно, не все студенты смогут посетить хотя бы k лекций. Поэтому они просят вас посчитать максимальное число студентов, которые смогут освоить курс программирования.

Формат входных данных

В первой строке входных данных содержится пять целых чисел:

- t — количество студентов;
- n — количество лекций;
- a — количество мест в первой аудитории;
- b — количество мест во второй аудитории;
- k — минимальное число лекций, которые нужно посетить, чтобы освоить курс программирования.

Ограничения: $1 \leqslant t, n, a, b, k \leqslant 10^9$.

Формат выходных данных

В единственной строке выходных данных выведите одно целое число — максимальное число студентов, которые смогут посетить хотя бы k лекций и, следовательно, освоить курс программирования.

Примеры

стандартный ввод	стандартный вывод
10 3 4 4 3	4
10 3 4 4 5	0
100000 100000 100000 100000 1	100000
5 4 5 3 3	5
100 9 6 3 6	7

Замечание

В четвертом примере 5 студентов могут освоить курс. Рассмотрим один из возможных вариантов:

1. На первую лекцию приходят студенты 1, 2, 3, 4, 5.
2. На вторую лекцию приходят студенты 1, 3.
3. На третью лекцию приходят студенты 1, 2, 3, 4, 5.
4. На четвертую лекцию приходят студенты 2, 4, 5.

Итого: каждый из пяти студентов посетил три лекции.

Задача M. Приятная прогулка

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Вдоль дороги, на которой живёт Аня, стоят n домов, каждый из которых раскрашен в один из k цветов.

Аня любит гулять вдоль дороги, но ей не нравится, когда подряд стоят два дома, раскрашенных в один и тот же цвет. Она хочет выбрать такой участок для прогулки, вдоль которого никакие два соседних дома не раскрашены в один цвет.

Помогите Ане найти участок дороги, содержащий максимальное число домов, вдоль которого ей будет приятно гулять.

Формат входных данных

Первая строка ввода содержит два целых числа n и k — количество домов и количество цветов ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant k \leqslant 100\,000$).

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ — цвета домов вдоль дороги ($1 \leqslant a_i \leqslant k$).

Формат выходных данных

Выведите одно число — максимальное количество домов на участке дороги, вдоль которого Ане приятно гулять.

Пример

стандартный ввод	стандартный вывод
8 3 1 2 3 3 2 1 2 2	4

Замечание

В примере максимальный приятный для Ани участок дороги идёт от 4 до 7 дома, дома на нём раскрашены в цвета $[3, 2, 1, 2]$, он содержит 4 дома.