Задача A. Удвоение

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Алёна играет с калькулятором. Изначально на экране калькулятора находится целое положительное число n. Алёна хочет добиться того, чтобы на экране калькулятора оказалось число m. Для этого она может несколько (возможно, ноль или один) раз применить операцию удвоения числа, располагающегося на экране. Удастся ли девочке удвоениями получить из числа n число m?

Формат входных данных

В первой строке задано целое положительное число n ($1 \leqslant n \leqslant 100$) — число, изначально находящееся на экране калькулятора. Во второй строке задано целое положительное число m ($1 \leqslant m \leqslant 100$) — число, которое Алёна хочет получить.

Формат выходных данных

Выведите «`Yes`» (без кавычек), если m можно получить из n, применив операцию удвоения несколько (возможно, ноль или один) раз. В противном случае выведите «`No`» (без кавычек).

Вы можете выводить каждую букву в любом регистре.

Примеры

стандартный ввод	стандартный вывод
3 12	Yes
12 3	No
7 7	Yes
7 10	No

Пояснения к примерам

Число 12 можно получить из числа 3 двукратным применением операции удвоения, так как $3 \cdot 2 \cdot 2 = 12$, поэтому на первый пример ответ «`Yes`».

Число 3 нельзя получить из числа 12 последовательным применением операции удвоения несколько раз, так как 12 уже больше, чем 3, а применение операции удвоения только увеличивает число. Поэтому ответ на второй пример — «`No`».

Число 7 можно получить из числа 7, ни разу не применив операцию удвоения, поэтому ответ на третий пример тоже «`Yes`».

Наконец, как несложно убедиться, 10 нельзя получить из 7 последовательным применением операции удвоения, поэтому ответ на четвёртый пример — «`No`».

Система оценки

Задача содержит две подзадачи. Для того, чтобы решение было принято на проверку, необходимо, чтобы оно проходило **первый** тест из условия. Баллы за каждую подзадачу будут начислены, если пройдены все тесты этой и всех предыдущих подзадач.

Подзадача	Баллы	Ограничения	
		n	m
1	30	$1 \leqslant n \leqslant 2$	$1 \leqslant m \leqslant 2$
2	70	$1 \leqslant n \leqslant 100$	$1 \leqslant m \leqslant 100$

Задача B. До гор

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Петя и Вася играют в игру, в которой нужно составлять слова из плиток со строчными латинскими буквами. Для тренировки внимательности они ввели правило: в составляемых ими словах не должно быть подстроки «or», другими словами, запрещено иметь в слове две соседних буквы, из которых первая в нём «o», а вторая — «r».

Вася уже долго никак не мог обыграть Петю, как вдруг ему подвернулся шанс: Пете, когда он составил очередное слово, позвонили, и он отвлёкся от игры. Вася решил воспользоваться этим и сжульничать — поменять местами две плитки с буквами, чтобы в слове образовалась подстрока «or». При этом он не хочет, чтобы обман был раскрыт, поэтому он собирается сжульничать *незаметно* — поменять местами две *соседние* плитки. Кроме того, Вася может не жульничать, то есть ничего не менять в Петином слове.

Определите, может ли Вася добиться того, чтобы, когда Петя вернулся к игре, в его слове была подстрока «or», и при этом тот не обнаружил вмешательства.

Формат входных данных

В первой строке находится одно целое число n — длина строки, составленной Петей ($1 \leqslant n \leqslant 100\,000$). Во второй строке находится сама строка из n строчных латинских букв.

Формат выходных данных

Выведите «Yes», если Вася может получить в Петиной строке подстроку «or», поменяв местами два соседних символа или ничего не сделав. В противном случае выведите «No».

Вы можете выводить каждую букву в любом регистре (заглавную или строчную).

Примеры

стандартный ввод	стандартный вывод
3 orc	Yes
3 rio	No
5 narod	Yes
6 opqrst	No
1 r	No
2 og	No
3 ban	No

Пояснения к примерам

В первом примере Вася должен ничего не сделать, чтобы победить. Во втором, четвёртом, пятом, шестом и седьмом примерах, что бы Вася ни делал, получить подстроку «or» не получится. В третьем же примере надо поменять местами символы «r» и «o», чтобы они образовали нужную подстроку в правильном порядке.

Система оценки

Задача содержит три подзадачи. Баллы за каждую из подзадач начисляются при условии прохождения всех тестов этой подзадачи, тестов из условия, подходящих под её ограничения, и всех предыдущих подзадач.

Подзадача	Баллы	Ограничения
		n
1	20	$n \leqslant 3$
2	50	$n \leqslant 100$
3	30	$n \leqslant 10^5$

Задача C. Саша и задача про предков

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Деревом называется связный ациклический неориентированный граф. Подвесить дерево за вершину v — это значит задать на рёбрах ориентацию по следующей процедуре. Сначала берутся все смежные с вершиной v рёбра и направляются от неё. После этого аналогичная процедура повторяется для всех вершин v', смежных с v, затем для всех вершин v'', смежных с какой-либо из вершин v', и так далее. При этом однажды получившее ориентацию ребро не меняет её. Вершина v при этом называется корнем дерева.

Определим глубину вершины как расстояние от неё до корня. Для множества вершин S наименьшим общим предком (обозначим его $\mathcal{A}(S)$) называется вершина с наибольшей глубиной, из которой можно дойти до любой из вершин множества S по ориентированным рёбрам. В частности, наименьшим общим предком множества из одной вершины является сама эта вершина. Заметим, что вершина $\mathcal{A}(S)$ может не принадлежать множеству S.

Саша купил в магазине дерево из n вершин и собрался решать на нём задачи про наименьших общих предков. Задача, которую Саша хочет решить — это найти минимальное число k, для которого выполнено следующее условие: у любого множества S из k и более вершин $\mathcal{A}(S)$ — это корень дерева. Но вот незадача: в магазине забыли подвесить дерево. Чуть-чуть поразмыслив, Саша решил просто рассмотреть все n вариантов подвешивания. Помогите ему решить задачу для всех этих вариантов.

Формат входных данных

В первой строке находится число n — число вершин в дереве ($1 \leqslant n \leqslant 100\,000$).

Каждая из следующих $n - 1$ строк содержит по два целых числа u и v ($1 \leqslant u, v \leqslant n$, $u \neq v$), описывающих соответствующее ребро.

Гарантируется, что заданный граф образует дерево.

Формат выходных данных

Выведите n целых чисел, i-е из которых — это ответ на задачу, если корнем является вершина i.

Примеры

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

Пояснения к примерам

Рассмотрим первый пример. Если дерево подвешено за вершину 1, то рёбра ориентированы как $1 \to 2 \to 3$. Рассмотрим соответствующие множества и их наименьших общих предков:

- $\mathcal{A}(\{1\}) = 1$
- $\mathcal{A}(\{2\}) = 2$
- $\mathcal{A}(\{3\}) = 3$
- $\mathcal{A}(\{1, 2\}) = 1$
- $\mathcal{A}(\{1, 3\}) = 1$
- $\mathcal{A}(\{2, 3\}) = 2$
- $\mathcal{A}(\{1, 2, 3\}) = 1$

Несложно видеть, что в таком случае минимальное k будет равно 3.

Теперь переподвесим дерево за вершину 2. В таком случае рёбра будут выглядеть как $2 \to 1$ и $2 \to 3$.

- $\mathcal{A}(\{1\}) = 1$
- $\mathcal{A}(\{2\}) = 2$
- $\mathcal{A}(\{3\}) = 3$
- $\mathcal{A}(\{1, 2\}) = 2$
- $\mathcal{A}(\{1, 3\}) = 2$
- $\mathcal{A}(\{2, 3\}) = 2$
- $\mathcal{A}(\{1, 2, 3\}) = 2$

Здесь наименьший общий предок любого двухэлементного множества является корнем дерева. Следовательно, ответ равен 2.

Подвешивание дерева за вершину 3 аналогично подвешиванию за вершину 1.

Система оценки

Задача содержит четыре подзадачи. Баллы за каждую из подзадач начисляются только при условии прохождения всех тестов этой подзадачи и всех тестов предыдущих подзадач.

Группа	Баллы	Ограничения	
		n	Дополнительно
1	15	$n \leqslant 15$	–
2	10	$n \leqslant 10^5$	Все рёбра имеют вид $(u, u + 1)$.
3	35	$n \leqslant 1000$	–
4	40	$n \leqslant 10^5$	–

Задача D. Коллективный пароль

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Андроид Геннадий управляет электронным банком. В этом банке каждый счёт защищён паролем, состоящим ровно из девяти маленьких английских букв.

Сегодня к Геннадию пришли новые клиенты — группа из n альдебаранцев. Они хотят открыть общий счёт, но не вполне доверяют друг другу. Альдебаранцы хотят, чтобы каждый из них по отдельности не мог управлять счётом, поэтому просят сделать им индивидуальные пароли, каждый из p маленьких английских букв, причём $p < 9$. Однако, если больше половины альдебаранцев из группы одновременно введут свои пароли, банк должен восстановить по ним общий девятибуквенный пароль и обеспечить доступ к счёту. Никаких других условий альдебаранцы не ставят.

Итак, Геннадию нужно добавить новый модуль к своему электронному банку. Этот модуль, получив девятибуквенный пароль, размер группы n и длину индивидуальных паролей p, должен генерировать n индивидуальных паролей длины p. А получив любые $\lceil \frac{n}{2} \rceil$ из этих индивидуальных паролей, он должен выдавать исходный девятибуквенный пароль.

Напишите программу, которая обеспечит работу нового модуля.

Система оценки

Задача содержит три подзадачи. Баллы за каждую из подзадач начисляются только при условии прохождения всех тестов этой подзадачи, а также примера из условия.

Подзадача	Баллы	Ограничения		
		n	$\lceil \frac{n}{2} \rceil$	p
1	30	$n = 3$	$\lceil \frac{n}{2} \rceil = 2$	$p = 7$
2	30	$n = 5$	$\lceil \frac{n}{2} \rceil = 3$	$p = 7$
3	40	$n = 7$	$\lceil \frac{n}{2} \rceil = 4$	$p = 4$

Протокол взаимодействия

В этой задаче ваше решение будет запущено на каждом тесте два раза. В каждом тесте может быть несколько тестовых случаев.

При первом запуске решение из девятибуквенных паролей делает индивидуальные пароли. В первой строке записано слово «`split`». Вторая строка содержит целые числа t, n и p — количество тестовых случаев, размер группы и длину индивидуальных паролей ($1 \leqslant t \leqslant 1000$). Каждая из следующих t строк описывает один тестовый случай и содержит девять маленьких английских букв — очередной пароль. В ответ на каждый тестовый случай решение должно вывести строку, содержащую n индивидуальных паролей, разделённых пробелами. Каждый индивидуальный пароль должен иметь длину p и состоять из маленьких английских букв.

При втором запуске решение по индивидуальным паролям восстанавливает девятибуквенные пароли. В первой строке записано слово «`merge`». Вторая строка содержит целые числа t, n и p — количество тестовых случаев, размер группы и длину индивидуальных паролей ($1 \leqslant t \leqslant 1000$). Эти числа такие же, как и при первом запуске. Далее программа жюри как-то переставляет t строк, выведенных решением при первом запуске, в каждой строке как-то переставляет n выведенных паролей и оставляет какие-то $\lceil \frac{n}{2} \rceil$ из них. Конкретные действия зафиксированы заранее в каждом тесте. Получившиеся t строк и даются решению в качестве t тестовых случаев. В ответ на каждый тестовый случай решение должно восстановить девятибуквенный пароль и вывести строку, содержащую этот пароль.

При всех запусках каждая строка входных данных, включая последнюю, завершается переводом строки.

Примеры

В каждом тесте входные данные при втором запуске зависят от того, что вывело решение при первом запуске. В примере мы рассмотрим решение, которое делает индивидуальные пароли так: первый пароль — это первые семь букв, второй — это последние семь букв, а третий — это семь букв "a". При восстановлении решение сначала пытается поставить первый полученный индивидуальный пароль на первые семь позиций, а второй — на последние семь позиций, после чего проверяет, совпали ли буквы на пяти позициях в середине. Если нет, решение меняет местами два полученных пароля, после чего опять делает то же самое. К сожалению, такое решение не всегда может восстановить пароль.

Далее показаны два запуска этого решения на первом тесте.

стандартный ввод	стандартный вывод
split 4 3 7 passwords uhaaaaaaa aaaaaaaaa plainword	passwor sswords aaaaaaa uhaaaaa aaaaaaa aaaaaaa aaaaaaa aaaaaaa aaaaaaa plainwo ainword aaaaaaa
merge 4 3 7 aaaaaaa aaaaaaa passwor sswords uhaaaaa aaaaaaa ainword plainwo	aaaaaaaaa passwords uhaaaaaaa plainword

Задача E. Лазерный луч

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Словно не замечая недоверчивых взглядов собравшихся вокруг зевак, Гриша, имея в качестве оборудования всего лишь медный подстаканник, разрядившуюся лазерную указку, пару батареек, скрепку, дробовик и стопятидесятиметровый циклотрон, неведомым образом собрал лазерный фонарик. Не в первый раз он удивлял всех, создавая невероятные работающие приборы из вещей, которые были в каждом доме, но при этом, казалось бы, по функционалу весьма уступали тому, что Гриша собирался соорудить. Теперь же он расхаживал по полю, водя фонариком из стороны в сторону и рассматривая причудливый узор, вырисовывавшийся ярким зелёным лучом на голом льду и грязи.

Вскоре Грише стало интересно, что будет, если направить свет фонарика идеально горизонтально. Встав в полюбившуюся ему точку P, Гриша выбрал другую точку Q на той же высоте и направил в неё свой фонарик. Гришина планета представляла собой бесконечную во все стороны плоскость, поэтому с большой высоты свет фонарика выглядел как луч из точки P в точку Q. Деревьев и высоких домов на планете почти что не было, поэтому свет луча ничего не прерывало, и его длина оказалась бесконечной.

Когда Гриша пришёл домой, он узнал в новостях, что разжёг небольшой политический скандал. Дело в том, что на его планете находилось несколько государств, и луч задел территорию некоторых из них; это вызвало споры о том, кому принадлежал свет этого луча. Некоторые политики утверждали, что владельцем являлось государство, на территории которого находился лазерный фонарик, испускавший луч, иные говорили, что права на свет фонарика должно было иметь из всех государств, задетых лучом, то, в котором проживало больше всего людей. Гриша не понимал, зачем все до сих пор продолжали так бурно обсуждать этот вопрос (особенно если учесть, что, поразвлекавшись вдоволь, Гриша выключил фонарик, и лазерный луч уже давно исчез), но объяснять общественности бессмысленность подобных споров ему не захотелось. Вместо этого Гриша решил придумать свой критерий, наиболее справедливый, по которому будет решено, чей же на самом деле свет луча.

Чтобы не терять времени даром, Гриша достал карту «Государства мира» и принялся её изучать. Выяснилось, что распределение территории планеты устроено очень просто. А именно, в каждом из государств есть принадлежащая ему точка, называемая *столицей*, а каждая из остальных точек планеты принадлежит тому государству, которому принадлежит ближайшая к ней столица. Формально, пусть на планете n государств, столицей i-го из которых является точка C_i; тогда точка A принадлежит i-му государству, если для каждого $j \in \{1, 2, \ldots, n\}$, не равного i, выполнено $AC_i < AC_j$. Если такого i нет, другими словами, если есть несколько ближайших к точке столиц (больше одной), то она считается граничной и не принадлежит ни одному из государств.

Гриша легко доказал, что при таком территориальном устройстве планеты верно следующее: начиная с определённого места, бесконечный кусок любого луча либо содержится в территории какого-то из государств, либо пролегает ровно по границе между государствами и ни в каком из них не лежит. Именно это свойство Гриша и решил использовать в качестве решающего фактора: если имеется государство, содержащее бесконечную часть луча, то оно и должно владеть его светом, а если же внутри государств луч не пролегает, или пролегает лишь его часть конечной длины, то луч считается ничьим. Гриша, поставив на карте метки, соответствующие точкам P и Q, провёл луч и легко определил владельца света фонарика. А вы справитесь?

Формат входных данных

В первой строке находится одно целое число n — количество государств ($1 \leqslant n \leqslant 100\,000$). В каждой из следующих n строк находится по два целых числа x_i, y_i, разделённых пробелом — координаты столицы i-го государства ($-10^9 \leqslant x_i, y_i \leqslant 10^9$). В каждой из двух следующих строк находится по два целых числа, разделённых пробелом — координаты x_P, y_P точки P и координаты x_Q, y_Q точки

Q соответственно ($-10^9 \leqslant x_P, y_P, x_Q, y_Q \leqslant 10^9$). Гарантируется, что местоположения всех столиц различны, а также что точки P и Q различны.

Формат выходных данных

Если луч принадлежит i-му государству (другими словами, если длина части луча на его территории бесконечна), выведите единственное целое число i ($1 \leqslant i \leqslant n$). Если же луч не принадлежит ни одному из государств, выведите -1.

Примеры

стандартный ввод	стандартный вывод
2 0 0 10 0 0 0 10 0	2
3 1 3 4 0 0 0 -3 -4 -1 -2	-1

Система оценки

Задача содержит шесть подзадач. Для того, чтобы решение было принято на проверку, необходимо, чтобы оно проходило **первый** тест из условия. Баллы за каждую из подзадач начисляются при условии прохождения всех тестов этой подзадачи, тестов из условия, подходящих под ограничения этой подзадачи, и всех предыдущих подзадач, ограничения в которых подходят под ограничения этой подзадачи.

Для удобства будем говорить, что луч PQ *совпадает с осью* Ox, если P — начало координат, а Q лежит на положительной части оси абсцисс на расстоянии от 1 до 100 от начала координат; формально, $x_P = y_P = y_Q = 0$ и $1 \leqslant x_Q \leqslant 100$.

Подзадача	Баллы	Ограничения	
		n	Координаты
1	5	$n = 2$	$0 \leqslant x_i, y_i \leqslant 100$, луч PQ совпадает с осью Ox
2	7	$n = 2$	$-100 \leqslant x_i, y_i \leqslant 100$, луч PQ совпадает с осью Ox
3	9	$n = 2$	$0 \leqslant x_i, y_i, x_P, y_P, x_Q, y_Q \leqslant 100$
4	23	$1 \leqslant n \leqslant 100\,000$	$-10^9 \leqslant x_i, y_i \leqslant 10^9$, луч PQ совпадает с осью Ox
5	25	$1 \leqslant n \leqslant 100$	$-10^4 \leqslant x_i, y_i, x_P, y_P, x_Q, y_Q \leqslant 10^4$
6	31	$1 \leqslant n \leqslant 100\,000$	$-10^9 \leqslant x_i, y_i, x_P, y_P, x_Q, y_Q \leqslant 10^9$

Задача F. Пончики

Входные данные:	*стандартный ввод*
Выходные данные:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Вы содержите небольшую, но очень известную в кругах истинных ценителей лавочку под названием R&B Donuts, которая продаёт красные и синие пончики. Как и у любого уважающего себя заведения, у вас есть определённое количество постоянных посетителей, вкусы которых вам хорошо известны.

В целях оптимизации продаж вы изучили привычки самых преданных клиентов и составили предварительный план, в котором содержится информация о том, сколько людей придёт за красными и синими пончиками в каждый из n следующих дней. Доподлинно известно, что для счастья каждому из клиентов требуется ровно один пончик его любимого цвета; в случае отсутствия такового клиент уйдёт из лавочки недовольным и больше не вернётся.

Секрет вашего успеха заключается в волшебном устройстве для изготовления пончиков, которое выпекает ровно k пончиков каждое утро. К сожалению, волшебство имеет свои пределы, поэтому все пончики, произведённые в один день, должны иметь один и тот же цвет. Пончики не портятся (они же волшебные!), поэтому их можно продавать не только в день приготовления, но и в любой день после.

Чему равно максимальное количество клиентов, которые уйдут из лавочки счастливыми в следующие n дней при условии оптимальной выпечки пончиков?

Формат входных данных

В первой строке заданы два числа n и k ($1 \leqslant n \leqslant 10^5$, $1 \leqslant k \leqslant 10^8$) — количество дней и объём ежедневной партии пончиков в штуках.

Следующие n строк содержат по два числа r_i и b_i ($0 \leqslant r_i, b_i \leqslant n \cdot k$) — количество посетителей в i-й день, желающих приобрести красные и синие пончики соответственно.

Формат выходных данных

Выведите одно число — максимальное количество покупателей, которые уйдут из лавочки довольными.

Примеры

стандартный ввод	стандартный вывод
2 10 10 9 10 9	20
2 10 9 7 0 13	20
3 1 0 1 1 0 1 1	3

Пояснения к примерам

В первом примере разумно и в первый, и во второй день готовить красные пончики.

Во втором примере в первый день следует приготовить 10 синих пончиков, а затем продать 7 из них. Во второй день также стоит выбрать 10 синих пончиков: с учётом трёх оставшихся синих пончиков с предыдущего дня, суммарные продажи составят $7 + 13 = 20$ штук.

В третьем примере в первый день стоит ориентироваться на синие пончики, во второй — на красные. Что касается третьего дня, то здесь оба варианта равнозначны.

Система оценки

Задача содержит пять подзадач. Баллы за каждую подзадачу начисляются, если пройдены все тесты этой подзадачи и всех предыдущих подзадач.

Подзадача	Баллы	Ограничения	
		n	k
1	7	$n \leqslant 15$	$k \leqslant 15$
2	17	$n \leqslant 30$	$k \leqslant 30$
3	26	$n \leqslant 1000$	$k \leqslant 100$
4	22	$n \leqslant 1000$	$k \leqslant 10^4$
5	28	$n \leqslant 10^5$	$k \leqslant 10^8$