Задача A. Сломанные роботы

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

У Кати есть n свечек, пронумерованных от 1 до n. Изначально они не горят.

Также у неё есть m роботов. i-й робот зажигает свечки $a_i, a_i + d_i, a_i + 2 \cdot d_i, \ldots, a_i + t \cdot d_i$, где t — наибольшое целое число такое, что $a_i + t \cdot d_i \leqslant n$. Катя хочет приказать всем роботам сделать свою работу. Свечка горит, если хотя бы один из роботов её зажёг.

К сожалению, ровно k роботов сломаны, но Катя не помнит, какие именно!

Найдите матожидание количества зажжённых свечек после того, как Катя прикажет всем роботам сделать свою работу, в предположении, что любой набор из k сломанных роботов равновероятен.

Это матожидание может быть представлено в виде p/q, где p и q — взаимно простые неотрицательные числа. Вы должны вывести $(p \cdot q^{-1}) \mod (10^9 + 7)$, где q^{-1} — обратный к q по модулю.

Формат входных данных

Первая строка содержит три целых числа n, m, k ($1 \leqslant n, m \leqslant 2 \cdot 10^5$, $1 \leqslant k \leqslant m$).

Следующие m строк содержат описание роботов. i-я из этих строк содержит два целых числа a_i и d_i ($1 \leqslant a_i, d_i \leqslant n$).

Формат выходных данных

Выведите одно число — ответ.

Пример

стандартный ввод	стандартный вывод
4 2 1 1 2 2 2	2

Замечание

В примере ровно один робот сломан.

Если сломан первый робот, то второй зажжёт свечки $2, 4$.

Если сломан второй робот, первый зажжёт свечки $1, 3$.

Таким образом, матожидание количества зажжёных свечек равно $\frac{1}{2} \cdot 2 + \frac{1}{2} \cdot 2 = \frac{2}{1}$ и $(2 \cdot 1^{-1}) \mod (10^9 + 7)$ равно 2.

Задача B. Разбиение

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3.5 секунд
Ограничение по памяти:	256 мегабайт

Ваня работает в поисковом отделе Очень Известной Компании. Каждый день его код обрабатывает миллионы запросов.

Программа, которую пишет Ваня, занимается обработкой запроса. А именно, он разбивает строку запроса на непересекающиеся подстроки-токены. Каждый из токенов должен быть подстрокой строки t. При этом, хотелось бы, чтобы токены в разбиении были большими (то есть токены отражали смысл поискового запроса). А именно, длина минимального токена должна быть максимальной.

Сегодня на сервер пришел поисковый запрос, состоящий из строки s, и Ванин код перестал работать, упав по Undefined behavior. К сожалению, ответить на запрос нужно прямо сейчас, поэтому Ваня просит вас помочь ему. Реализуйте его программу.

Формат входных данных

На вход программе даются строки s, t ($1 \leqslant |s|,\ |t| \leqslant 10^5$). Строки состоят из букв латинского алфавита, причем могут быть как заглавными, так и строчными.

Формат выходных данных

Выведите максимально возможную длину минимальной по длине подстроки разбиения. Если искомого разбиения не существует, выведите «1».

Пример

стандартный ввод	стандартный вывод
zachettop chtoproishoditzachto	1

Задача C. Монополия

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

В Тридесятом государстве есть N фирм, занимающихся разработкой программного обеспечения. Однажды известный олигарх Тридесятого государства Иванушка решил монополизировать эту отрасль. Для этого он хочет купить максимальное число программистских фирм Тридесятого государства.

Он разослал предложения всем N компаниям и через некоторое время получил от каждой их них согласие или отказ. Однако он знает, что в бизнесе очень многое зависит от взаимного доверия партнеров.

В результате небольшого исследования Иванушка установил, между какими компаниями существует взаимное доверие (причем всегда если компания доверяет компании B, то компания B доверяет компании A).

Теперь, при желании, Иванушка может повторно разослать предложения всем компаниям, включив в письма список компаний, давших согласие участвовать в его проекте. При этом каждая компания, независимо от своего первоначального мнения дает согласие, если в списке есть хотя бы одна компания, которой она доверяет, и отказ в противном случае. Таким образом, некоторые компании, которые изначально не согласились участвовать в проекте, могут теперь дать свое согласие, а некоторые из давших согласие — наоборот отказаться. В результате этого у Иванушки формируется новый список, который он опять может разослать фирмам. Он может сколь угодно долго повторять операцию, каждый раз рассылая текущий список. Иванушка может остановить процесс в любой момент и заключить договора с теми, кто после последней рассылки дал согласие.

Напишите программу, которая определит, какое максимальное число компаний может объединить Иванушка под своим началом.

Будем считать, что Иванушка — честный предприниматель и он никогда не подтасовывает рассылаемые им списки.

Формат входных данных

В первой строке входных данных содержится число N — количество фирм ($1 \leqslant N \leqslant 2000$). Далее идут N чисел, описывающих ответ фирмы на первое предложение Иванушки (1 — согласие, 0 — отказ). Далее задается число M ($0 \leqslant M \leqslant 200000$) — количество пар компаний, между которыми существует доверие. Далее следуют M пар чисел, задающих номера фирм, между которыми существует взаимное доверие (числа в паре не могут быть одинаковыми). Любая пара компаний упоминается в этом списке не более одного раза.

Формат выходных данных

Выведите одно число — максимальное число фирм, которое сможет купить Иванушка.

Пример

стандартный ввод	стандартный вывод
7 1 0 0 0 0 0 1 6 1 2 1 3 1 4 4 5 5 6 2 5	4

Задача D. Программирование квадрокоптеров

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Школьники готовятся к участию в соревновании по программированию квадрокоптеров. Квадрокоптер, который используется в соревновании, может выполнять две команды: подняться вверх на 1 метр и опуститься вниз на 1 метр. Команда подъёма обозначается символом «`(`», а команда спуска — символом «`)`».

Программа для квадрокоптера представляет собой последовательность команд. Программа считается корректной, если, начав её исполнение на уровне земли и выполнив последовательно все команды, квадрокоптер снова оказывается на уровне земли. При этом в процессе выполнения программы квадрокоптер не должен пытаться опуститься ниже уровня земли.

Например, следующие программы являются корректными: «`()(())`», «`((()))`». Программа«`(((`» не является корректной, поскольку квадрокоптер завершает ее выполнение на высоте 3 метра над уровнем земли, программа «`())(`» также не является корректной, поскольку при выполнении третьей команды квадрокоптер пытается опуститься ниже уровня земли.

Участник соревнования написал корректную программу для квадрокоптера, состоящую из n команд, пронумерованных от 1 до n. Он загрузил её в память квадрокоптера для демонстрации во время соревнования. К сожалению, после загрузки программы в память квадрокоптера участник случайно удалил её на своём компьютере, а квадрокоптер не позволяет выгрузить программу из своей памяти.

К счастью, квадрокоптер поддерживает специальный режим отладки программы. В этом режиме квадрокоптер с загруженной в него программой может отвечать на специальные запросы. Каждый запрос представляет собой два целых числа: l и r, $1 \leqslant l \leqslant r \leqslant n$. В ответ на запрос квадрокоптер сообщает, является ли фрагмент загруженной в него программы, состоящий из команд с l-й по r-ю включительно, корректной программой для квадрокоптера, либо нет. Участник хочет с помощью режима отладки восстановить загруженную в квадрокоптер программу.

Требуется написать программу-решение, которая взаимодействует с программой жюри, моделирующей режим отладки квадрокоптера, и в итоге восстанавливает загруженную в квадрокоптер программу.

Протокол взаимодействия

Это интерактивная задача.

Сначала на вход подаётся целое число n — количество команд в программе квадрокоптера ($2 \leqslant n \leqslant 50\,000$).

Для каждого теста жюри зафиксировано число k — максимальное количество запросов. Гарантируется, что k запросов достаточно, чтобы решить задачу. Это число не сообщается программе-решению. Ограничения k в различных подзадачах приведены в таблице системы оценивания. Если программа-решение делает более k запросов к программе жюри, то на этом тесте она получает в качестве результата тестирования «Неверный ответ».

Чтобы сделать запрос, программа-решение должна вывести строку вида «`?` l r», где l и r — целые положительные числа, задающие фрагмент программы квадрокоптера ($1 \leqslant l \leqslant r \leqslant n$).

В ответ на запрос программы-решения программа жюри подаёт ей на вход либо строку «`Yes`», либо строку «`No`», в зависимости от того, является ли запрошенный фрагмент программы квадрокоптера корректной программой.

Если программа-решение определила ответ на задачу, то она должна вывести строку «`!` $c_1c_2\ldots c_n$», где символ c_i задаёт i-ю команду в программе квадрокоптера и равен либо «`(`», либо «`)`».

После этого программа-решение должна завершиться.

Гарантируется, что в каждом тесте программа в памяти квадрокоптера является фиксированной корректной программой, которая не меняется в зависимости от запросов, произведённых программой-решением.

Система оценки

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	k		
1	21	$2 \leqslant n \leqslant 16$	$k = 150$		Потестовые
2	28	$2 \leqslant n \leqslant 100$	$k = 10\,000$	1	Потестовые
3	26	$2 \leqslant n \leqslant 1000$	$k = 10\,000$	1, 2	Потестовые
4	25	$2 \leqslant n \leqslant 50\,000$	$k = 100\,000$	1 – 3	Потестовые

Примеры

стандартный ввод	стандартный вывод
4 Yes No Yes Yes	 ? 1 4 ? 1 3 ? 1 2 ? 3 4 ! ()()
6 Yes No Yes	 ? 3 4 ? 1 2 ? 2 5 ! ((()))

Замечание

Приведённые примеры иллюстрируют взаимодействие программы-решения с программой жюри «по шагам», для чего в них добавлены дополнительные пустые строки. При реальном тестировании лишние пустые строки вводиться не будут, выводить пустые строки также не требуется.

В первом примере $n = 4$. Единственная возможная корректная программа из двух команд это «()», поэтому из результатов третьего и четвёртого запросов можно сделать вывод, что программа в памяти квадрокоптера — «()()». Поэтому, в частности, ответ на второй запрос действительно «No», так как фрагмент программы «()(» не является корректной программой: если квадрокоптер исполнит именно эти три команды, он останется на уровне одного метра над землёй.

В втором примере $n = 6$, и произведённых запросов достаточно, чтобы однозначно определить, что программа в памяти квадрокоптера — «((()))».

В тестах из условия $k = 150$, то есть, разрешается произвести не более 150 запросов.

Задача F. Разрезы прямоугольника

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Прямоугольник со сторонами A и B разрезали на прямоугольники разрезами, параллельными его сторонам. Например, если было сделано p горизонтальных и q вертикальных разрезов, то было получено $(p+1) \cdot (q+1)$ прямоугольников. В результате было получено n различных видов прямоугольников. Два прямоугольника различны, если хотя бы одна сторона у них имеет разную длину. Обратите внимание, что прямоугольники не поворачивали, то есть прямоугольники $a \times b$ и $b \times a$ считаются различными при $a \neq b$.

Для каждого вида прямоуольников даны размеры прямоугольников этого вида, а также количество прямоугольников этого вида, которое было получено в результате разреза исходного прямоугольника.

Посчитайте количество пар $(A; B)$ таких, что в результате разрезания прямоугольника со сторонами A и B могли быть получены данные прямоугольники. Обратите внимание, что $(A; B)$ и $(B; A)$ считаются различными парами при $A \neq B$.

Формат входных данных

В первой строке находится целое число n ($1 \leqslant n \leqslant 2 \cdot 10^5$) — количество видов прямоугольников, полученных в результате разрезания исходного прямоугольника.

В следующих n строках содержатся три целых числа w_i, h_i, c_i ($1 \leqslant w_i, h_i, c_i \leqslant 10^{12}$) — размеры прямоугольника некоторого вида и количество прямоугольников такого вида.

Гарантируется, что прямоугольники разных видов различны.

Формат выходных данных

Выведите одно целое число — ответ на задачу.

Примеры

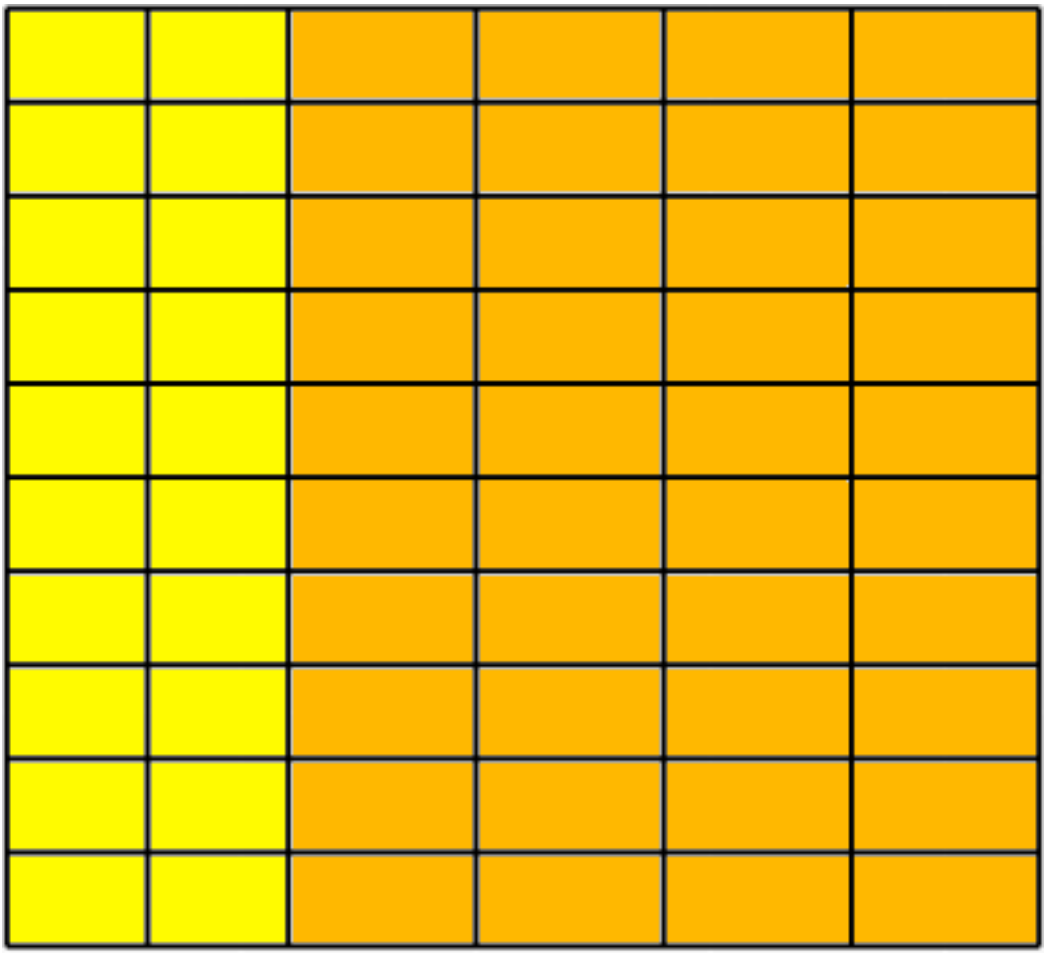
стандартный ввод	стандартный вывод
1 1 1 9	3
2 2 3 20 2 4 40	6
2 1 2 5 2 3 5	0

Замечание

В первом примере подходящими парами являются $(1; 9)$, $(3; 3)$ и $(9; 1)$.

Во втором примере есть 6 подходящих пар: $(2; 220)$, $(4; 110)$, $(8; 55)$, $(10; 44)$, $(20; 22)$ и $(40; 11)$.

Ниже пример разрезания для получения $(20; 22)$.



В третьем примере не существует ни одной подходящей пары.

Задача G. Кубики

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.25 секунд
Ограничение по памяти:	256 мегабайт

Родители подарили Пете набор детских кубиков. Поскольку Петя скоро пойдет в школу, они купили ему кубики с буквами. На каждой из шести граней каждого кубика написана буква.

Теперь Петя хочет похвастаться перед старшей сестрой, что научился читать. Для этого он хочет сложить из кубиков её имя. Но это оказалось довольно сложно сделать — ведь разные буквы могут находиться на одном и том же кубике и тогда Петя не сможет использовать обе буквы в слове. Правда одна и та же буква может встречаться на разных кубиках. Помогите Пете!

Дан набор кубиков и имя сестры. Выясните, можно ли выложить её имя с помощью этих кубиков и если да, то в каком порядке следует выложить кубики.

Формат входных данных

В первой строке вводится число N ($1 \leqslant N \leqslant 100$) — количество кубиков в наборе у Пети. Во второй строке задано имя Петиной сестры — слово, состоящие только из больших латинских букв, не длиннее 100 символов. Следующие N строк содержат по 6 букв (только большие латинские буквы), которые написаны на соответствующем кубике.

Формат выходных данных

В первой строке выведите «`YES`» если выложить имя Петиной сестры данными кубиками можно, «`NO`» в противном случае.

В случае положительного ответа, во второй строке выведите M различных чисел из диапазона от 1 до N, где M — количество букв в имени Петиной сестры. i-е число должно быть номером кубика, который следует положить на i-е место при составлении имени Петиной сестры. Кубики нумеруются с 1, в том порядке, в котором они заданы во входных данных. Если решений несколько, выведите любое. Разделяйте числа пробелами.

Примеры

стандартный ввод	стандартный вывод
2 AB AAAAAB AAAAAA	YES 2 1
3 ANNY AAAAAA NNNNNN YYYYYY	NO

Задача H. Опять скобки

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вам дается строка s, состоящая из символов ‘(’ и ‘)’. От вас требуется найти количество позиций i таких, что при удалении из строки s символа s_i оставшаяся строка является правильной скобочной подпоследовательностью.

Правильной скобочной подпоследовательностью называется такой набор скобок, который можно получить, выкинув из корректного математического выражения все символы, кроме скобок. Более формально:

- Пустая строка является правильной скобочной последовательностью.
- Если S_1, S_2 — правильные скобочные последовательности, то "S_1S_2-— правильная скобочная последовательность.
- Если S_1 — правильная скобочная последовательность, то "(S_1)-— правильная скобочная последовательность.

Формат входных данных

В первой строке дается строка s ($2 \leqslant |s| \leqslant 5 \cdot 10^5$). Строка состоит только из символов ‘(’ и ‘)’.

Формат выходных данных

Выведите одно число — количество способов получить из s правильную скобочную последовательность удалением ровно одного символа.

Примеры

стандартный ввод	стандартный вывод
()()())	4
()((())()	4

Замечание

В первом примере можно удалить любую закрывающую скобку.

Задача I. Мега-инверсии

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.25 секунд
Ограничение по памяти:	256 мегабайт

Инверсией в перестановке $p_1, p_2, \ldots, p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовём мега-инверсией в перестановке $p_1, p_2, \ldots, p_N$ тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Напишите алгоритм для быстрого подсчёта количества мега-инверсий в перестановке.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, \ldots, p_N$ ($1 \leqslant p_i \leqslant N$), все p_i попарно различны. Числа разделяются переводами строк.

Формат выходных данных

Единственная строка выходного файла должна содержать одно число, равное количеству мега-инверсий в перестановке $p_1, p_2, \ldots, p_N$.

Пример

стандартный ввод	стандартный вывод
4 4 3 2 1	4

Задача J. И снова сумма...

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Реализуйте структуру данных, которая поддерживает множество S целых чисел, с котором разрешается производить следующие операции:

- $add(i)$ — добавить в множество S число i (если он там уже есть, то множество не меняется);
- $sum(l, r)$ — вывести сумму всех элементов x из S, которые удовлетворяют неравенству $l \leqslant x \leqslant r$.

Формат входных данных

Исходно множество S пусто. Первая строка входного файла содержит n — количество операций ($1 \leqslant n \leqslant 300\,000$). Следующие n строк содержат операции. Каждая операция имеет вид либо «`+` i», либо «`?` l r». Операция «`?` l r» задает запрос $sum(l, r)$.

Если операция «`+` i» идет во входном файле в начале или после другой операции «`+`», то она задает операцию $add(i)$. Если же она идет после запроса «`?`», и результат этого запроса был y, то выполняется операция $add((i + y) \bmod 10^9)$.

Во всех запросах и операциях добавления параметры лежат в интервале от 0 до 10^9.

Формат выходных данных

Для каждого запроса выведите одно число — ответ на запрос.

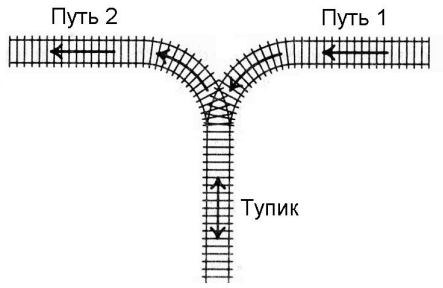
Пример

стандартный ввод	стандартный вывод
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4	3 7

Задача K. Сортировка вагонов

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

К тупику со стороны пути 1 (см. рисунок) подъехал поезд. Разрешается отцепить от поезда один или сразу несколько первых вагонов и завезти их в тупик (при желании, можно даже завезти в тупик сразу весь поезд). После этого часть из этих вагонов вывезти в сторону пути 2. После этого можно завезти в тупик еще несколько вагонов и снова часть оказавшихся вагонов вывезти в сторону пути 2. И так далее (так, что каждый вагон может лишь один раз заехать с пути 1 в тупик, а затем один раз выехать из тупика на путь 2). Заезжать в тупик с пути 2 или выезжать из тупика на путь 1 запрещается. Нельзя с пути 1 попасть на путь 2, не заезжая в тупик.



Известно, в каком порядке изначально идут вагоны поезда. Требуется с помощью указанных операций сделать так, чтобы вагоны поезда шли по порядку (сначала первый, потом второй и т.д., считая от головы поезда, едущего по пути 2 в сторону от тупика).

Формат входных данных

Вводится число N — количество вагонов в поезде ($1 \leqslant N \leqslant 2000$). Дальше идут номера вагонов в порядке от головы поезда, едущего по пути 1 в сторону тупика. Вагоны пронумерованы натуральными числами от 1 до N, каждое из которых встречается ровно один раз.

Формат выходных данных

Если сделать так, чтобы вагоны шли в порядке от 1 до N, считая от головы поезда, когда поезд поедет по пути 2 из тупика, можно, выведите действия, которые нужно проделать с поездом. Каждое действие описывается двумя числами: типом и количеством вагонов:

- если нужно завезти с пути 1 в тупик K вагонов, должно быть выведено сначала число 1, а затем — число $K (K \geqslant 1)$,
- если нужно вывезти из тупика на путь 2 K вагонов, должно быть выведено сначала число 2, а затем — число $K (K \geqslant 1)$.

Если возможно несколько последовательностей действий, приводящих к нужному результату, выведите любую из них.

Если выстроить вагоны по порядку невозможно, выведите одно число 0.

Примеры

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

Задача L. Нелёгкие подпоследовательности [на 10]

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дано n строк из маленьких английских букв. Рассмотрим все перестановки этих строк. Определите, для скольких из них строка, полученная конкатенацией строк, переставленных в соответствии с перестановкой, имеет чётное число различных подпоследовательностей.

Формат входных данных

Первая строка содержит целое число n ($2 \leqslant n \leqslant 20$).

Следующие n строк содержат строки из маленьких английских букв. Длины каждой строки не превосходит 10^5.

Формат выходных данных

Выведите единственное число — ответ.

Примеры

стандартный ввод	стандартный вывод
2 a ba	1
3 a a baa	4

Замечание

Рассмотрим первый пример.

У строки "baa" = "ba" + "a" различных подпоследовательностей 6: пустая, "a", "b", "aa", "ba", "baa".

У строки "aba" = "a" + "ba" различных подпоследовательностей 7: пустая, "a", "b", "aa", "ab", "ba", "aba".

Поэтому ответ 1.

Задача M. LCA-3

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

> *Подвешенное дерево* —это ориентированный граф без циклов, в котором в каждую вершину, кроме одной, называемой *корнем* ориентированного дерева, входит одно ребро. В корень ориентированного дерева не входит ни одного ребра. *Отцом* вершины называется вершина, ребро из которой входит в данную.

(по материалам Wikipedia)

Дан набор подвешенных деревьев. Требуется выполнять следующие операции:

1. 0 u v Для двух заданных вершин u и v выяснить, лежат ли они в одном дереве. Если это так, вывести вершину, являющуюся их наименьшим общим предком, иначе вывести 0.
2. 1 u v Для корня u одного из деревьев и произвольной вершины v другого дерева добавить ребро (v, u). В результате эти два дерева соединятся в одно.

Вам необходимо выполнять все операции online, т.е. вы сможете узнать следующий запрос только выполнив предыдущий.

Формат входных данных

На первой строке входного файла находится число n — суммарное количество вершин в рассматриваемых деревьях, $1 \leqslant n \leqslant 50000$. На следующей строке расположено n чисел — предок каждой вершины в начальной конфигурации, или 0, если соответстующая вершина является корнем. Затем следует число k — количество запросов к вашей программе, $1 \leqslant k \leqslant 100000$. Каждая из следующих строк содержит по три целых числа: вид запроса (0 — для поиска LCA или 1 — для добавления ребра) и два числа x, y. Вершины, участвующие в запросе можно вычислить по формуле: $u = (x - 1 + ans) \bmod n + 1$, $v = (y - 1 + ans) \bmod n + 1$, где ans - ответ на последний запрос типа 0 ($ans = 0$ для первого запроса).

Формат выходных данных

Для каждого запроса типа 0, выведите в выходной файл одно число на отдельной строке — ответ за этот запрос.

Пример

стандартный ввод	стандартный вывод
5 0 0 0 0 0 12 1 5 3 0 2 5 1 4 2 1 1 5 0 1 5 1 3 4 0 1 5 0 3 1 0 4 2 0 1 4 0 5 2 0 4 1	0 5 5 3 2 3 3 2

Задача N. Дерево и запросы

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Задано корневое дерево, состоящее из n вершин. Каждая вершина дерева имеет определенный цвет. Будем считать, что вершины дерева пронумерованы целыми числами от 1 до n. Тогда цвет вершины v будем обозначать c_v. Корнем дерева является вершина с номером 1.

В задаче вам требуется ответить на m запросов. Каждый запрос характеризуется двумя целыми числами v_j, k_j. Ответ на запрос v_j, k_j – это количество таких цветов вершин x, что в поддереве вершины v_j содержится как минимум k_j вершин цвета x.

Формат входных данных

В первой строке записаны два целых числа n и m ($2 \leqslant n \leqslant 10^5$; $1 \leqslant m \leqslant 10^5$). В следующей строке записана последовательность целых чисел c_1, c_2, …, c_n ($1 \leqslant ci \leqslant 10^5$). В следующих $n - 1$ строках записаны ребра дерева. В i-ой строке записаны числа a_i, b_i ($1 \leqslant a_i, b_i \leqslant n$; $a_i \neq b_i$) – вершины, соединенные ребром дерева.

Далее в m строках записаны запросы. В j-той строке записаны два целых числа v_j, k_j ($1 \leqslant v_j \leqslant n$; $1 \leqslant k_j \leqslant 10^5$).

Формат выходных данных

Выведите m целых чисел – ответы на запросы в порядке появления запросов во входных данных.

Пример

стандартный ввод	стандартный вывод
8 5 1 2 2 3 3 2 3 3 1 2 1 5 2 3 2 4 5 6 5 7 5 8 1 2 1 3 1 4 2 3 5 3	2 2 1 0 1

Задача O. Петербург?

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

– Это что за остановка –
Бологое иль Поповка? –
А с платформы говорят:
– Это город Ленинград.

«Вот какой рассеянный», Самуил Маршак

Пытаясь спастись от мира спортивного программирования, Алина сбежала на вокзал и уехала прочь на ночной электричке. Минуты медленно уплывали в даль, и уставшую девочку клонило в сон. Ей снился город-сказка, где не надо программировать, а можно гулять, мечтать и наслаждаться жизнью. Внезапно дождь из **интерактивных** задач разрушил эту идиллию.

Проснувшись и открыв окно, Алина задалась вопросом весьма философского свойства: «Где я?». С перрона потерявшейся девочке сообщили, что этот город, не похожий ни на что вокруг, представляет собой неориентированный граф на n вершинах и m ребрах. Сей невероятный факт, однако, нисколько не удивил Алину. Она давно мечтала побывать в одном таком городе — Петербурге. Его уникальной отличительной особенностью является то, что хотя бы **половина** его ребер — мосты (определение дано в конце условия). Так как никакие другие города Алине не интересны, она решила ограничиться расспросом находящихся на платформе эрудированных путешественников. Любой из их них может по данной вершине v сообщить любое ещё не названное ребро, исходящее из нее, или же заявить об отсутствии таковых.

Алина неуверена в своих силах, поэтому попросила вас помочь ей определить, попала ли она в Петербург. Так как её поезд скоро продолжит свой путь, задать больше $3n$ вопросов не получится.

Обратите внимание, что в графе **могут** присутствовать петли и кратные ребра.

Протокол взаимодействия

В первой строке стандартного потока ввода даны два целых числа n и m ($1 \leqslant n, m \leqslant 100\,000$) — число вершин и ребер в графе соответственно.

Для того, чтобы узнать очередное ребро, исходящее из u-й вершины ($1 \leqslant u \leqslant n$), нужно вывести «`?` u». После этого ваша программа на вход получит целое число v ($-2 \leqslant v \leqslant -1$ или $1 \leqslant v \leqslant n$) — $v = a + b - u$, если существует ребро ab, которое инцидентно вершине u и **ещё не было названо**, -1, если такого ребра не существует и -2, если вы превысили допустимое число запросов. В последнем случае ваша программа должна немедленно завершиться, в ином случае жюри не гарантирует корректность полученного вами вердикта.

Вам разрешается задать не более $3n$ вопросов.

Чтобы сообщить, что ответ найден, требуется вывести «`! Yes`» или «`! No`», в зависимости от того, является ли загаданный граф Петербургом. В случае положительного ответа выведите $\lceil \frac{m}{2} \rceil$ строк, по два целых числа u_i и v_i в каждой ($1 \leqslant u_i, v_i \leqslant n$), обозначающих, что ребро (u_i, v_i) является мостом. Любое ребро в приведенном списке должно встречаться не более одного раза (кратные ребра считаются различными).

Запрос на вывод ответа не входит в ограничение на $3n$ запросов.

Не забывайте сбрасывать буфер после каждого запроса. Например, на языке C++ надо использовать функцию `fflush(stdout)` или вызов `cout.flush()`, на Java вызов `System.out.flush()`, на Pascal `flush(output)` и `stdout.flush()` для языка Python.

Примеры

стандартный ввод	стандартный вывод
3 3 2 2 -1 3 -1 -1	 ? 3 ? 1 ? 2 ? 1 ? 1 ? 3 ! No
4 4 2 3 2 -1 4 -1 -1 -1	 ? 1 ? 2 ? 3 ? 1 ? 3 ? 3 ? 2 ? 4 ! Yes 1 2 3 4

Замечание

В условии в примере взаимодействия вводимые и выводимые данные расположены для удобства восприятия в хронологическом порядке, при реальном взаимодействии никакие «лишние» переводы строк возникать не должны.

Ввод-вывод в примерах демонстрирует пример взаимодействия вашей программы с проверяющей системой.

В первом примере был загадан граф на трех вершинах с ребрами $(1, 2)$, $(2, 3)$ и $(3, 1)$.

Во втором примере была загадан граф на четырех вершинах с ребрами $(1, 2)$, $(2, 3)$, $(3, 4)$ и $(2, 3)$.

Ребро, соединяющее вершины u и v, называется мостом, если после его удаления между вершинами u и v не существует пути.