Задача A. Минимальное остовное дерево

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан связный неориентированный взвешенный граф. Необходимо выбрать в этом графе некоторое подмножество рёбер минимального суммарного веса таким образом, чтобы между любыми двумя вершинами графа существовал путь из выбранных рёбер.

Очевидно, что выбранное подмножество рёбер должно быть деревом (для минимальности суммарного веса ребер), и такое подмножество называется *минимальным остовным деревом* или *минимальным каркасом*.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($1 \leqslant n \leqslant 20\,000$, $0 \leqslant m \leqslant 100\,000$). Следующие m строк содержат описание рёбер по одному на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — номера концов ребра и его вес соответственно ($1 \leqslant b_i, e_i \leqslant n$, $0 \leqslant w_i \leqslant 100\,000$).

Гарантируется, что данный граф является связным.

Формат выходных данных

Программа дожна вывести одно целое число — вес минимального остовного дерева.

Примеры

stdin	stdout
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача B. Разрезание графа

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф. Над ним в заданном порядке производят операции следующих двух типов:

- `cut` — разрезать граф, то есть удалить из него ребро;
- `ask` — проверить, лежат ли две вершины графа в одной компоненте связности.

Известно, что после выполнения всех операций типа `cut` рёбер в графе не осталось. Найдите результат выполнения каждой из операций типа `ask`.

Формат входных данных

Первая строка входного файла содержит три целых числа, разделённые пробелами — количество вершин графа n, количество рёбер m и количество операций k ($1 \leqslant n \leqslant 50\,000$, $0 \leqslant m \leqslant 100\,000$, $m \leqslant k \leqslant 150\,000$).

Следующие m строк задают рёбра графа; i-я из этих строк содержит два числа u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$), разделённые пробелами — номера концов i-го ребра. Вершины нумеруются с единицы; граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции. Операция типа `cut` задаётся строкой «`cut` u v» ($1 \leqslant u, v \leqslant n$), которая означает, что из графа удаляют ребро между вершинами u и v. Операция типа `ask` задаётся строкой «`ask` u v» ($1 \leqslant u, v \leqslant n$), которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности. Гарантируется, что каждое ребро графа встретится в операциях типа `cut` ровно один раз.

Формат выходных данных

Для каждой операции `ask` во входном файле выведите на отдельной строке слово «`YES`», если две указанные вершины лежат в одной компоненте связности, и «`NO`» в противном случае. Порядок ответов должен соответствовать порядку операций `ask` во входном файле.

Пример

stdin	stdout
3 3 7 1 2 2 3 3 1 ask 3 3 cut 1 2 ask 1 2 cut 1 3 ask 2 1 cut 2 3 ask 3 1	YES YES NO NO

Задача C. Реструктуризация компании

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

В жизни даже самой успешной компании может наступить кризисный период, когда приходится принимать тяжёлое решение о реструктуризации, распускать и объединять отделы, увольнять работников и заниматься прочими неприятными делами. Рассмотрим следующую модель компании.

В Большой Софтверной Компании работают n человек. Каждый человек принадлежит какому-то *отделу*. Исходно каждый человек работает над своим проектом в своём собственном отделе (таким образом, в начале компания состоит из n отделов по одному человеку).

Однако, в жизни компании наступили тяжёлые времена, и руководство было вынуждено нанять кризисного менеджера, который начал переустраивать рабочий процесс для повышения эффективности производства. Обозначим за $team(person)$ команду, в которой работает человек $person$. Кризисный менеджер может принимать решения двух типов:

1. Объединить отделы $team(x)$ и $team(y)$, сформировав из них один большой отдел, содержащий всех сотрудников $team(x)$ и $team(y)$, где x и y ($1 \leqslant x, y \leqslant n$) — номера каких-то двух сотрудников компании. Если $team(x)$ совпадает с $team(y)$, ничего делать не требуется.

2. Объединить отделы $team(x), team(x+1), \ldots, team(y)$, где x и y ($1 \leqslant x \leqslant y \leqslant n$) — номера каких-то двух сотрудников компании.

При этом кризисный менеджер иногда может интересоваться, работают ли в одном отделе сотрудники x и y ($1 \leqslant x, y \leqslant n$).

Помогите кризисному менеджеру, ответив на все его запросы.

Формат входных данных

Первая строка входных данных содержит два целых числа n и q ($1 \leqslant n \leqslant 200\,000$, $1 \leqslant q \leqslant 500\,000$) — количество сотрудников компании и количество запросов кризисного менеджера.

В последующих q строках находятся запросы кризисного менеджера. Каждый запрос имеет вид $type\ x\ y$, где $type \in \{1, 2, 3\}$. Если $type = 1$ или $type = 2$, то запрос представляет собой решение кризисного менеджера об объединении отделов соответственно первого или второго вида. Если $type = 3$, то требуется определить, работают ли в одном отделе сотрудники x и y. Обратите внимаие, что x может равняться y в запросе любого типа.

Формат выходных данных

На каждый запрос типа 3 выведите «`YES`» или «`NO`» (без кавычек), в зависимости от того, работают ли в одном отделе соответствующие люди.

Примеры

стандартный ввод	стандартный вывод
8 6 3 2 5 1 2 5 3 2 5 2 4 7 2 1 2 3 1 7	NO YES YES

Задача D. Ещё одна задача про деревья

Имя входного файла:	`trees.in`
Имя выходного файла:	`trees.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Есть граф из N вершин. Изначально он пустой. Нужно обработать M запросов:

1. добавить ребро из вершины v_1 в вершину v_2, при этом вершины v_1 и v_2 находятся в разных деревьях и вершина v_2 является корнем какого-то дерева.
2. по двум вершинам a и b определить, лежат ли они в одном дереве.

Решение задачи: реализуем СНМ с эвристикой сжатия путей:

```
int n, m, l[NMAX];

int calc_leader (int v) {
    if (l[v] != v)
        l[v] = calc_leader (l[v]);
    return l[v];
}

int main() {
    scanf ("%d%d", &n, &m);
    for (int i = 1; i <= n; i ++)
        l[i] = i;
    for (int i = 1; i <= m; i ++) {
        int x, y, z;
        scanf ("%d%d%d", &z, &x, &y);
        if (z == 1)
            l[y] = x;
        else if (calc_leader (x) == calc_leader (y))
            printf ("YES\n");
        else
            printf ("NO\n");
    }
}
```

Вам же предстоит сделать тест, на котором это решение будет работать долго. Более точно, нужно сделать тест, на котором количество вызовов функции `calc_leader` будет не меньше, чем $\frac{1}{4} M \log_2 M$.

Формат входных данных

Входной файл содержит два целых числа N и M. В тестах, отличных от примера $N = 1000000, M = 100000$.

Формат выходных данных

Выведите M строк. i-ая строка должна иметь вид `1` x y, если i-ый запрос заключается в добавлении ребра из вершины x в вершину y, или `0` x y, если спрашивается, лежат ли вершины x и y в одном дереве.

Примеры

trees.in	trees.out
2 2	1 2 1 0 1 1

Замечание

На тест из примера будет зачтен любой ответ, который удовлетворяет формату вывода.

Задача E. Всем чмоки в этом чатике!

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мегабайт

Сегодня Мэри, как программисту социальной сети «Телеграфчик», предстоит реализовать сложную систему управления чатами.

Задача Мэри усложняется тем, что в социальную сеть «Телеграфчик» внедрена продвинутая система шифрования «ZergRus», простая, как всё гениальное. Суть её в том, что в системе хранится одна переменная zerg, которая принимает значения от 0 (включительно) до $p = 10^6 + 3$ (исключая p) и меняется в зависимости от событий в системе.

В социальной сети всего n пользователей ($1 \leqslant n \leqslant 10^5$). В начале дня каждый пользователь оказывается в своём собственном чате, в котором больше никого нет. Переменная zerg в начале дня устанавливается равной 0.

В течение дня происходят события типов:

1. Участник с номером $(i + zerg) \bmod n$ посылает сообщение всем участникам, сидящим с ним в чате (в том числе и себе самому), при этом переменная zerg заменяется на $(30 \cdot zerg + 239) \bmod p$.

2. Происходит слияние чатов, в которых сидят участники с номерами $(i + zerg) \bmod n$ и $(j + zerg) \bmod n$. Если участники и так сидели в одном чате, то ничего не происходит. Если в разных, то чаты объединяются, а переменной zerg присваивается значение $(13 \cdot zerg + 11) \bmod p$.

3. Участник с номером $(i + zerg) \bmod n$ хочет узнать, сколько сообщений он не прочитал, и прочитать их. Если участник прочитал q новых сообщений, то переменной zerg присваивается значение $(100\,500 \cdot zerg + q) \bmod p$.

Вы поможете Мэри реализовать систему, обрабатывающую эти события?

Формат входных данных

В первой строке входного файла записаны натуральные числа n ($1 \leqslant n \leqslant 10^5$) — число пользователей социальной сети. и m ($1 \leqslant m \leqslant 3 \cdot 10^5$) — число событий, произошедших за день. В следующих m строках содержится описание событий. Первое целое число в строке означает тип события t ($1 \leqslant t \leqslant 3$). Если $t = 1$, далее следует число i ($0 \leqslant i < n$), по которому можно вычислить, какой участник послал сообщение. Если $t = 2$, далее следуют числа i и j ($0 \leqslant i, j < n$), по которым можно вычислить номера участников, чаты с которыми должны объединиться. Если $t = 3$, далее следует число i ($0 \leqslant i < n$), по которому можно вычислить номер участника, желающего узнать, сколько у него сообщений, и прочитать их.

Формат выходных данных

Для каждого события типа 3 нужно вывести число непрочитанных сообщений у участника.

Примеры

stdin	stdout	Пояснение
4 10 1 0 1 2 1 1 1 2 3 1 2 1 2 1 3 3 3 2 3 2 3 2	1 1 2	4 10 1 0 1 1 1 2 1 3 3 0 2 0 1 1 1 3 0 2 2 1 3 1

Задача F. День Объединения

Имя входного файла:	`unionday.in`
Имя выходного файла:	`unionday.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В Байтландии есть целых n городов, но нет ни одной дороги. Король страны, Вальдемар де Беар, решил исправить эту ситуацию и соединить некоторые города дорогами так, чтобы по этим дорогам можно было добраться от любого города до любого другого. Когда строительство будет завершено, король планирует отпраздновать День Объединения. К сожалению, казна Байтландии почти пуста, поэтому король требует сэкономить деньги, минимизировав суммарную длину всех построенных дорог.

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 5\,000$) — количество городов в Байтландии. Каждая из следующих n строк содержит по два целых числа x_i, y_i — координаты i-го города ($-10\,000 \leqslant x_i, y_i \leqslant 10\,000$). Никакие два города не расположены в одной точке.

Формат выходных данных

Первая строка выходного файла должна содержать минимальную суммарную длину дорог. Выведите число с точностью не менее 10^{-3}.

Пример

unionday.in	unionday.out
6 1 1 7 1 2 2 6 2 1 3 7 3	9.6568542495

Задача G. Дерево минимального ора

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

В последнее время Влад увлёкся остовными деревьями, так что его друзья не долго думая подарили ему на день рождения связный взвешенный неориентированный граф из n вершин и m рёбер.

Влад определил *орность* остовного дерева как побитовое ИЛИ всех его весов и теперь его интересует, какова минимальная возможная *орность*, которой можно добиться, выбрав некоторое остовное дерево. Остовное дерево — связный подграф данного графа, не содержащий циклов.

Иными словами вы хотите оставить $n - 1$ ребро, так чтобы граф остался связным и побитовое ИЛИ весов рёбер было как можно меньше. Вы должны найти само побитовое ИЛИ.

Формат входных данных

В первой строке входных данных записано целое число t ($1 \leqslant t \leqslant 10^4$) — количество наборов входных данных в тесте.

Перед каждым набором в тесте записана пустая строка.

Далее следуют два числа n и m ($3 \leqslant n \leqslant 2 \cdot 10^5, n - 1 \leqslant m \leqslant 2 \cdot 10^5$) — количество вершин и рёбер графа, соответственно.

Следующие m строк содержат описание рёбер. Строка i содержит три числа v_i, u_i и w_i ($1 \leqslant v_i, u_i \leqslant n$, $1 \leqslant w_i \leqslant 10^9$, $v_i \neq u_i$) — вершины, которые соединяет ребро, и его вес.

Гарантируется, что сумма m и сумма n по всем наборам входных данных не превосходит $2 \cdot 10^5$ и каждый тест содержит связный граф.

Формат выходных данных

Выведите t строк, каждая из которых содержит ответ на соответствующий набор входных данных — минимальную возможную *орность* остовного дерева.

Пример

стандартный ввод	стандартный вывод
3 3 3 1 2 1 2 3 2 1 3 2 5 7 4 2 7 2 5 8 3 4 2 3 2 1 2 4 2 4 1 2 1 2 2 3 4 1 2 1 2 3 2 1 3 3 3 1 4	2 10 3

Задача H. Подсчет опыта

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В очередной онлайн игре игроки, как обычно, сражаются с монстрами и набирают опыт. Для того, чтобы сражаться с монстрами, они объединяются в кланы. После победы над монстром, всем участникам клана, победившего его, добавляется одинаковое число единиц опыта. Особенностью этой игры является то, что кланы никогда не распадаются и из клана нельзя выйти. Единственная доступная операция — объединение двух кланов в один.

Поскольку игроков стало уже много, вам поручили написать систему учета текущего опыта игроков.

Формат входных данных

В первой строке входного файла содержатся числа n ($1 \leqslant n \leqslant 200\,000$) и m ($1 \leqslant m \leqslant 200\,000$) — число зарегистрированных игроков и число запросов.

В следующих m строках содержатся описания запросов. Запросы бывают трех типов:

- `join X Y` — объединить кланы, в которые входят игроки `X` и `Y` (если они уже в одном клане, то ничего не меняется).
- `add X V` — добавить `V` единиц опыта всем участникам клана, в который входит игрок `X` ($1 \leqslant V \leqslant 100$).
- `get X` — вывести текущий опыт игрока `X`.

Изначально у всех игроков 0 опыта и каждый из них состоит в клане, состоящим из него одного.

Формат выходных данных

Для каждого запроса `get X` выведите текущий опыт игрока `X`.

Примеры

стандартный ввод	стандартный вывод
3 6 add 1 100 join 1 3 add 1 50 get 1 get 2 get 3	150 0 50

Задача I. Элементы в рядах

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Игроки играют в игру «Элементы в рядах». Игра проходит на поле размера $A \cdot B \cdot C$. Игроки в каком-то порядке захватывают клетки поля. Побеждает игрок, который первым получает непрерывный ряд из k его клеток.

Во время игры участники забыли, какое число k они выиграли. Однако они записывали все ходы, которые делали в процессе игры. Теперь им интересно узнать для каждого хода, какое максимальное количество непрерывных клеток подряд было у игрока, который только что сделал ход.

Рядом считается любой прямой или диагональный набор клеток. Формально говоря, существуют ряды трёх типов:

1. Прямой ряд в плоскости, все клетки которого имеют одинаковые две координаты, а изменение третьей координаты для любой пары соседних клеток составляет 1 по модулю.

2. Диагональный ряд в плоскости, все клетки которого имеют одинаковую одну координату, а изменение двух других координат для любой пары соседних клеток составляет 1 по модулю, причем каждая из этих двух координат для любых пар соседних клеток $i - 1$ и i или только уменьшается на 1, или только увеличивается на 1.

3. Диагональный ряд в пространстве, все три координаты для любой пары соседних клеток которого отличаются на 1 по модулю. Каждая координата для любых пар соседних клеток $i - 1$ и i или только уменьшается на 1, или только увеличивается на 1.

Более формально, рядом считается любая последовательность клеток, для которой существуют стартовая клетка ряда (x, y, z) и направление (dx, dy, dz) такие, что $|dx|, |dy|, |dz| \leqslant 1$ и i-я клетка ряда имеет координаты $(x + i \cdot dx, y + i \cdot dy, z + i \cdot dz)$ (нумерация i с нуля).

Формат входных данных

В первой строке входных данных находятся числа A, B, C, m и p — размеры поля, количество ходов, сделанных в игре, и количество игроков ($1 \leqslant A, B, C \leqslant 120000$, $1 \leqslant m \leqslant 100000$, $1 \leqslant p \leqslant 10$). Гарантируется, что $A \cdot B \cdot C \leqslant 120000$.

В следующих m строках описаны сами ходы. Каждый ход задаётся числами i, x, y и z ($1 \leqslant i \leqslant p$, $1 \leqslant x \leqslant A$, $1 \leqslant y \leqslant B$, $1 \leqslant z \leqslant C$).

Формат выходных данных

Для каждого хода, необходимо вывести максимальное количество непрерывных клеток подряд в каком-либо ряду для игрока, который только что совершил ход.

Пример

стандартный ввод	стандартный вывод
3 4 5 10 3 1 1 1 1 2 2 1 2 1 2 2 2 1 3 3 3 2 2 3 2 2 2 4 2 2 3 4 5 3 3 1 3 3 2 2 3 3 1 3 3	1 1 2 3 1 2 2 1 2 3

Задача J. Нефтяное дело

Имя входного файла:	`oil.in`
Имя выходного файла:	`oil.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам дан неориентированный связный граф из n вершин и m рёбер. Для каждого ребра известна стоимость его удаления в тугриках. У вас есть s тугриков. Вы хотите удалить как можно больше рёбер так, чтобы граф оставался связным, а суммарная стоимость всех удалённых рёбер не превосходила s тугриков.

Формат входных данных

Первая строка входных данных содержит три целых числа n, m и s — количесто вершин в графе, количество рёбер в графе и количество тугриков соответственно ($2 \leqslant n \leqslant 50\,000$, $1 \leqslant m \leqslant 100\,000$, $0 \leqslant s \leqslant 10^{18}$). Следующие m строк содержат описания рёбер графа. Каждое описание состоит из трёх целых чисел — номера вершин, которые соединяет данное ребро, и стоимость удаления ребра в тугриках (стоимость не превосходит 10^9). В графе не бывает кратных рёбер и петель.

Формат выходных данных

В первый строке выведите максимальное количество удаляемых рёбер. Во второй строке выведите номера удаляемых рёбер (рёбра нумеруются с единицы в порядке, данном во входном файле).

Примеры

oil.in	oil.out
6 7 10 1 2 3 1 3 3 2 3 3 3 4 1 4 5 5 5 6 4 4 6 5	2 1 6

Задача K. Локальная сеть

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Лёха работает системным администратором в локальной сети. Его сеть соединяет множество квартир и располагается в нескольких зданиях.

Сеть постоянно расширяется и Лёхе поручено проложить новый участок сети. У него есть схема, на которой указаны все возможные соединения между парами квартир и для каждого соединения он знает длину провода, необходимого для его прокладки. Его цель состоит в том, чтобы все квартиры были подключены к сети (возможно через другие квартиры).

Компания, в которой работает Лёха покупает кабель только в одном специализированном магазине. В магазине продается кабель пятой и шестой категорий по цене P_5 и P_6 рублей за метр. При этом в наличии имеется только Q_5 метров кабеля пятой категории и Q_6 метров кабеля шестой категории.

Лёхе необходимо составить план постройки сети с наименьшими затратами. План представляет собой список соединений между квартирами, при этом каждому соединению должно быть приписано, кабель какой категории будет проложен между этими квартирами (пятой или шестой). Стоимость прокладки этой сети равна сумме стоимости прокладки всех соединений. Общая длина кабеля каждой категории не должна превышать количество кабеля, имеющегося в магазине.

Формат входных данных

В первой строке входного файла содержится число N — количество квартир, которые необходимо соединить и M — количество возможных соединений ($1 \leqslant N \leqslant 1000$, $1 \leqslant M \leqslant 10\,000$).

Следующие M строк содержат описание возможных соединений. Каждое описание состоит из трех чисел A, B и L — где A и B задают номера квартир, а L — длина соединения между ними ($1 \leqslant L \leqslant 100$). Квартиры занумерованы от 1 до N.

Последняя строка входного файла содержит числа P_5, Q_5, P_6, Q_6 — цену и количество кабеля пятой и шестой категории соответственно ($1 \leqslant P, Q \leqslant 10\,000$) .

Формат выходных данных

Если все квартиры можно соединить в сеть, то следует вывести N строк, описывающих план сети. Первая строка должна содержать стоимость прокладки сети. Следующие $N-1$ строк должны содержать описание соединений, представленных двумя числами каждое: A_i и C_i, где A_i — номер соединения в списке возможных соединений (от 1 до M), а C_i задает категорию кабеля и может принимать значения 5 или 6. Если планов несколько — выведите любой из них.

Если все квартиры соединить невозможно выведите слово `Impossible`.

Пример

стандартный ввод	стандартный вывод
6 7 1 2 7 2 6 5 1 4 8 2 3 5 3 4 5 5 6 6 3 5 3 2 11 3 100	65 1 6 5 6 4 5 2 5 7 6

Задача L. Галактический апокалипсис

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

Давным-давно в одной далекой-далекой галактике, было N планет. Также было $N-1$ межпланетных магистралей, соединявших между собой все планеты (не обязательно напрямую). Иными словами, сеть планет и магистралей образовывала дерево. Кроме того, каждая магистраль имеет свой показатель интересности, заданный неотрицательным целым числом. Пара планет (A , B) называется скучной, если выполняются следующие условия:

1. A и B - различные планеты.
2. В действующей сети межпланетных магистралей существует путь между A и B.
3. Побитовый XOR показателей интересности всех магистралей в этом пути равен 0.

Ныне в галактике правит злой император, и он планирует использовать Силу, чтобы уничтожить все межпланетные магистрали в определенном порядке. Для того, чтобы спасти вселенную от гибели, вам необходимо определить количество пар скучных планет и после каждого разрушения вновь подсчитывать эту величину.

Формат входных данных

Первая строка содержит одно целое число N ($1 \leqslant N \leqslant 100000$). Каждая из следующих $N-1$ строк содержит три целых числа A_i, B_i, Z_i ($1 \leqslant A_i, B_i \leqslant 100000, 0 \leqslant Z_i \leqslant 1000000000$), которые означают, что планеты с номерами A_i и B_i соединены магистралью с показателем интересности Z_i. Последняя строка содержит $N-1$ число: перестановку натуральных чисел от 1 до $N-1$, отражающую порядок уничтожения магистралей (если i-е число в строке равно j, то император уничтожит дорогу между планетами A_j и B_j на i-м шаге).

Формат выходных данных

Выведите N строк, в k-й строке выведите одно число — количество пар скучных планет после уничтожения $k-1$ дорог.

Примеры

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