Задача A. Разрезание графа

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

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 входного файла

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

Следующие m строк задают рёбра графа; i-ая из этих строк содержит два числа u_i и v_i ($1 \le u_i, v_i \le n$), 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" ($1 \le u, v \le n$), которая означает, что из графа удаляют ребро между вершинами u и v. Операция типа `ask` задаётся строкой "`ask` u v" ($1 \le u, v \le n$), 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 выходного файла

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

cutting.in	cutting.out
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 B. Вперёд!

Имя входного файла:	`movetofront.in`
Имя выходного файла:	`movetofront.out`
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мегабайт

Капрал Дукар любит раздавать приказы своей роте. Самый любимый его приказ — "Вперёд!". Капрал строит солдат в ряд и отдаёт некоторое количество приказов, каждый из них звучит так: "Рядовые с l_i по l_j — вперёд!"

Перед тем, как Дукар отдал первый приказ, солдаты были пронумерованы от 1 до n, слева направо. Услышав приказ "Рядовые с l_i по l_j — вперёд!", солдаты, стоящие на местах с l_i по l_j включительно, продвигаются в начало ряда, в том же порядке, в котором были.

Например, если в какой-то момент солдаты стоят в порядке $2, 3, 6, 1, 5, 4$, то после приказа "Рядовые с 2 по 3 — вперёд!", порядок будет таким: $3, 6, 1, 2, 5, 4$. А если потом Капрал вышлет вперёд солдат с 3 по 4, то порядок будет уже таким: $1, 2, 3, 6, 5, 4$.

Вам дана последовательность из приказов Капрала. Найдите порядок, в котором будут стоять солдаты после исполнения всех приказов.

Формат входного файла

В первой строке входного файла указаны числа n и m ($2 \le n \le 100\,000$, $1 \le m \le 100\,000$) — число солдат и число приказов. Следующие m строк содержат приказы в виде двух целых чисел: l_i и r_i ($1 \le l_i \le r_i \le n$).

Формат выходного файла

Выведите в выходной файл n целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

Пример

movetofront.in	movetofront.out
6 3 2 4 3 5 2 2	1 4 5 2 3 6

Задача C. Республики

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

В стране есть N городов, причём некоторые соединены двусторонними дорогами. Естественно, от любого города до любого доехать можно. Кроме того, для поддержания казны часть дорог была закрыта. Казначейство с удовольствием бы закрыло и ещё, однако тогда король не смог бы доехать до некоторых городов своей страны. Так что, увы и ах. В целях дальнейшей экономии был разработан план — разбить страну на республики и поручить заботу о дорогах местным правителям. Однако республики нельзя создавать абы как — если в ней будет меньше K дорог, то республиканская милиция не сможет оперативно справляться с беспорядками. Кроме того, конечно, в республике должно быть можно доехать от любого города до любого, не выезжая за её пределы. При этом казначейство хочет открыть как можно больше республик, считая, что чем больше республик, тем больше денег. Понятно, что каждый город необходимо поместить ровно в одну республику. Помогите ему это сделать.

Формат входного файла

В первой строке содержится два целых числа $1 \le N \le 100$ и $M = N - 1$ — соответственно количество городов и количество дорог в стране. В следующих M строках содержится описание дорог — каждая строка содержит два числа $1 \le x, y \le N$. В последней строке содержится число $1 \le K \le M$ — минимальное количество дорог в республике.

Формат выходного файла

В первой строке выведите R — максимальное число республик. Следующие R строк должны содержать описание республик, по одному описанию на строке. В каждом описании вначале выведите количество городов в республике, потом перечислите номера городов в этой республике.

Пример

republics.in	republics.out
5 4 1 2 3 2 2 4 4 5 2	1 5 1 2 3 4 5
6 5 1 2 3 2 2 4 4 5 5 6 2	2 3 1 2 3 3 4 5 6