Задача A. Минимальное остовное дерево

Имя входного файла:	`mst.in`
Имя выходного файла:	`mst.out`
Ограничение по времени:	0.5 секунда
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

mst.in	mst.out
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача B. Разрезание графа

Имя входного файла:	`cutting.in`
Имя выходного файла:	`cutting.out`
Ограничение по времени:	2 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

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 C. 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 D. 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 E. Yet another set merging

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Вам задано n различных чисел от 1 до n. Изначально каждое число лежит в своём множестве. Обозначим множество, которому принадлежит элемент x как $set(x)$.

Ваша задача поддерживать три вида операций:

1. Объединить множества $set(x)$ и $set(y)$, где $1 \leqslant x, y \leqslant n$. Если $set(x)$ совпадает с $set(y)$, ничего делать не требуется.
2. Объединить множества $set(x),\ set(x+1),\ \ldots,\ set(y)$, где $1 \leqslant x \leqslant y \leqslant n$.
3. Ответить на вопрос, правда ли, что числа x и y лежат в одном множестве ($1 \leqslant x, y \leqslant n$).

Формат входных данных

Первая строка входных данных содержит два целых числа n и m ($1 \leqslant n \leqslant 200000$, $1 \leqslant m \leqslant 500000$) — количество чисел и количество запросов.

В последних q строках находятся запросы. Каждый запрос имеет вид $type\ x\ y$, где $type \in \{1, 2, 3\}$. Обратите внимание, 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, находятся ли числа в одном множестве.

Примеры

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

Задача F. Уничтожение массива

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

Вам дан массив, состоящий из n неотрицательных целых чисел $a_1, a_2, \ldots, a_n$.

В этом массиве один за другим зачёркиваются числа. Вам задана перестановка чисел от 1 до n — порядок, в котором это происходит.

После зачёркивания очередного числа вам необходимо найти в этом массиве подотрезок с максимальной суммой, не содержащий ни одного зачёркнутого числа. Сумму чисел в пустом подотрезке считайте равной 0.

Формат входных данных

В первой строке входных данных записано число n ($1 \leqslant n \leqslant 100\,000$) — длина массива.

В второй строке записаны n целых чисел $a_1, a_2, \ldots, a_n$ ($0 \leqslant a_i \leqslant 10^9$).

В третьей строке входных данных записана перестановка чисел от 1 до n — порядок, в котором зачеркиваются числа.

Формат выходных данных

В выходной файл выведите n строк, каждая из которых должна содержать одно число — максимальную сумму на подотрезке заданного массива, не содержащем зачёркнутых чисел, после выполнения очередного действия.

Примеры

стандартный ввод	стандартный вывод
4 1 3 2 5 3 4 1 2	5 4 3 0
5 1 2 3 4 5 4 2 3 5 1	6 5 5 1 0
8 5 5 4 4 6 6 5 5 5 2 8 7 1 3 4 6	18 16 11 8 8 6 6 0

Замечание

В первом тестовом примере происходит следующее:

1. Зачеркивается третий элемент, массив принимает вид 1 3 $*$ 5. Отрезок с максимально суммой 5 состоит из одного числа 5.
2. Зачеркивается четвертый элемент, массив принимает вид 1 3 $*$ $*$. Отрезок с максимально суммой 4 состоит из двух чисел 1 3.
3. Зачеркивается первый элемент, массив принимает вид $*$ 3 $*$ $*$. Отрезок с максимально суммой 3 состоит из одного числа 3.

4. Зачеркивается оставшийся второй элемент, в этот момент непустых допустимых подотрезков не остается, поэтому здесь ответ равен нулю.

Задача G. Всем чмоки в этом чатике!

Имя входного файла:	`chat.in`
Имя выходного файла:	`chat.out`
Ограничение по времени:	2 секунды
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

chat.in	chat.out	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 H. 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.