

Задача А. 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 ($1 \leq n \leq 50\,000$, $0 \leq m \leq 100\,000$, $m \leq k \leq 150\,000$).

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

<code>cutting.in</code>	<code>cutting.out</code>
3 3 7	YES
1 2	YES
2 3	NO
3 1	NO
ask 3 3	
cut 1 2	
ask 1 2	
cut 1 3	
ask 2 1	
cut 2 3	
ask 3 1	

Задача В. Минимальный каркас

Имя входного файла: `mst.in`
Имя выходного файла: `mst.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Требуется найти в связном графе остовное дерево минимального веса.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($1 \leq n \leq 20\,000$, $0 \leq m \leq 100\,000$). Следующие m строк содержат описание ребер 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 ($1 \leq b_i, e_i \leq n$, $0 \leq w_i \leq 100\,000$).

Граф является связным.

Формат выходных данных

Выведите единственное целое число — вес минимального остовного дерева.

Примеры

<code>mst.in</code>	<code>mst.out</code>
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача С. 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 ($1 \leq n \leq 5000$) — количество городов в Байтландии. Каждая из следующих n строк содержит по два целых числа x_i, y_i — координаты i -го города ($-10000 \leq x_i, y_i \leq 10000$). 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

<code>unionday.in</code>	<code>unionday.out</code>
6 1 1 7 1 2 2 6 2 1 3 7 3	9.6568542495

Задача D. Масло

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

Между пунктами с номерами $1, 2, \dots, N$ ($N \leq 1500$) проложено несколько дорог. Длина каждой дороги известна. По этой системе дорог можно добраться из любого упомянутого пункта в любой другой. Автозаправки расположены только в пунктах. Требуется определить, какое максимальное расстояние без заправки должен быть в состоянии проезжать автомобиль, чтобы без проблем передвигаться между пунктами.

Формат входных данных

В первой строке входного файла находятся числа N и K (количество дорог), $1 \leq N \leq 1500$, $1 \leq K \leq 400\,000$. В следующих K строках указаны пары пунктов, связанных дорогами, и расстояние между ними — целое число километров, не превышающее 10 000.

Формат выходных данных

В выходном файле должно оказаться одно число — длина максимального пробега без дозаправки.

Примеры

<code>oil.in</code>	<code>oil.out</code>
3 2 1 2 5 1 3 10	10

Задача Е. Космическая экспедиция

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

В 2004 году обитатели планеты Кремонид организовали космическую экспедицию для полёта в соседнюю галактику, где по их расчётам существует планета, пригодная для жизни. На космическом корабле был сконструирован жилой комплекс, куда заселили множество ученых.

Жилой комплекс имеет форму прямоугольного параллелепипеда размера $n \times m \times k$. Комплекс разбит на кубические отсеки с размерами $1 \times 1 \times 1$, всего nmk отсеков. Каждый отсек имеет координаты (x, y, z) , соответствующие положению отсека в комплексе, где $1 \leq x \leq n, 1 \leq y \leq m, 1 \leq z \leq k$.

Расстоянием между двумя отсеками с координатами (x_1, y_1, z_1) и (x_2, y_2, z_2) назовём число $|x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|$. Два отсека находятся в одном ряду, если их координаты отличаются ровно одной компонентой (например, $(2, 4, 3)$ и $(2, 6, 3)$ находятся в одном ряду). Два отсека являются соседними, если расстояние между ними равно единице.

В каждый отсек был установлен персональный компьютер. После взлёта жители комплекса решили объединить свои компьютеры в сеть. Был разработан план прокладывания сети, который представляет собой следующую процедуру: выбираются два отсека, находящихся в одном ряду. Первый отсек назовём начальным, второй — конечным. Робот, прокладывающий сеть, стартует в начальном отсеке. На каждом шаге робот передвигается в тот соседний отсек, расстояние от которого до конечного минимально. При этом он соединяет пары компьютеров в соседних отсеках, через которые он проходит, если это не приводит к образованию цикла. Если же соединение приводит к образованию цикла, то робот запоминает координаты этой пары соседних отсеков и не соединяет компьютеры в них между собой. Робот перемещается, пока не достигнет конечного отсека.

Указанная процедура повторяется q раз.

Вам необходимо определить, какие пары отсеков запомнил робот.

Формат входных данных

Первая строка входного файла содержит четыре числа n, m, k, q ($2 \leq n, m, k \leq 100, 1 \leq q \leq 20\,000$).

Далее следует q строк, описывающих пары отсеков, между которыми продвигается робот. Каждая строка содержит шесть чисел: первые три числа — координаты начального отсека, оставшиеся три числа — координаты конечного отсека.

Формат выходных данных

Для каждой пары отсеков, которую робот запомнил, выходной файл должен содержать строку с шестью числами — координатами отсеков в порядке прохождения их роботом.

Пример

<code>expedition.in</code>	<code>expedition.out</code>
5 4 2 6	3 3 1 2 3 1
2 4 1 2 1 1	3 1 1 2 1 1
5 1 1 5 4 1	3 1 1 2 1 1
5 1 1 2 1 1	
5 3 1 1 3 1	
3 1 1 1 1 1	
3 1 1 2 1 1	

Задача F. 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 ($1 \leq n \leq 10^5$). 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 ($1 \leq n \leq 10^5$) — число пользователей социальной сети, и m ($1 \leq m \leq 3 \cdot 10^5$) — число событий, произошедших за день. В следующих m строках содержится описание событий. Первое целое число в строке означает тип события t ($1 \leq t \leq 3$). Если $t = 1$, далее следует число i ($0 \leq i < n$), по которому можно вычислить, какой участник послал сообщение. Если $t = 2$, далее следуют числа i и j ($0 \leq i, j < n$), по которым можно вычислить номера участников, чаты с которыми должны объединиться. Если $t = 3$, далее следует число i ($0 \leq i < n$), 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	4 10
1 0	1	1 0
1 2	2	1 1
1 1		1 2
1 2		1 3
3 1		3 0
2 1 2		2 0 1
1 3		1 1
3 3		3 0
2 3 2		2 2 1
3 2		3 1