Содержание

У вас не получается читать/выводить данные?
Воспользуйтесь примерами (c++) (python).

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Обратите внимание на `GNU C++` компиляторы с суффиксом `inc`.
Подни можно пользоваться дополнительной библиотекой (`optimization.h`).
То есть, использовать быстрый ввод-вывод: **пример про числа и строки**.
И быструю аллокацию памяти (ускоряет vector-set-map-весь-STL): пример.

Для тех, кто хочет разобраться, как всё это работает.
Короткая версия быстрого ввода-вывода (тык) и короткая версия аллокатора (тык).

Must have

Задача 21A. Остовное дерево 2 [0.1, 256]

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. 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 ($1 \leqslant b_i, e_i \leqslant n$, $0 \leqslant w_i \leqslant 100\,000$). $n \leqslant 20\,000, m \leqslant 100\,000$.

Граф является связным.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — вес минимального остовного дерева.

Примеры

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

Задача 21B. Ближе к предкам [0.1, 256]

Страна Древляндия изначально состояла из одного древнего города 0. Время от времени города некоторого города p_i, недовольные условиями жизни уезжали из p_i и создавали новый город. Каждый следующий город получал минимальный не использованный номер, то есть, i. Жители города i никогда не забывают, что их предки пришли именно из p_i. Последнее время города по программе «слияние с предками» стали объединяться в регионы. Время от времени город i объявляет «не должно быть больше никаких границ между нами и предками из города p_i, давайте объединим наши городские агломерации в одну агломерацию».

Вам, как министру экономики Древляндии интересно в каждый момент времени знать размер самой большой городской агломерации, количество городов, в неё входящее.

Формат входных данных

Входные данные состоят из одного или нескольких тестов. На первой строке число тестов t, далее t однотипных тестов, каждый из которых задан следующим образом.

На 1-й строке n ($2 \leqslant n \leqslant 10^5$). На 2-й строке числа $p_1, p_2, \ldots p_{n-1}$: $1 \leqslant p_i < i - \forall$ города номер города предков. На 3-й строке перестановка из $n-1$ числа от 1 до $n-1$ – порядок операций слияния, число x обозначает, что регионы городов x и p_x сливаются в один.

Сумма n по всем тестам также не превосходит 10^5.

Формат выходных данных

Для каждого теста выведите одну строку из $n-1$ числа, числа в этой строке – размеры максимальных регионов после каждой из операций слияния в тесте.

Пример

stdin	stdout
3 2 0 1 4 0 1 2 3 2 1 4 0 1 2 1 3 2	2 2 3 4 2 2 4

Обязательные задачи

Задача 21C. Разрезание графа [0.1, 256]

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-ая 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 21D. Больные вершины [0.1, 256]

Случилось страшное, в древнем великом дереве вершины начали заболевать. Вы пока не понимаете причину болезни, пытаетесь разобраться, для этого нужно уметь быстро узнавать ближайшую к i в направлении корня больную вершину.

Формат входных данных

Входные данные состоят из одного или нескольких тестов. На первой строке число тестов t, далее t однотипных тестов, каждый из которых задан следующим образом.

На 1-й строке число вершин в дереве n ($2 \leqslant n \leqslant 10^5$) и число запросов q ($1 \leqslant q \leqslant 10^5$). Корень дерева – вершина 1. На 2-й строке числа $p_2, p_3, \ldots p_n$: $1 \leqslant p_i < i$. p_v – отец вершины v в дереве. Изначально все вершины здоровы. Следующие q строк содержат заросы вида «? i» – найти ближайшую больную от i в направлении корня и «- i» – вершина i заболела.

Сумма n и q по всем тестам также не превосходит 10^5.

Формат выходных данных

Для каждого теста выведите одну строку, содержащую ответы на запросы вида «?». Если больных в направлении корня нет, ответом будет -1.

Пример

stdin	stdout
3 2 5 1 ? 2 - 1 ? 2 - 2 ? 2 3 4 1 1 - 2 ? 3 - 3 ? 1 6 4 1 2 3 4 5 - 2 ? 1 ? 2 ? 6	-1 1 2 -1 -1 -1 2 2

Задача 21E. Ребра добавляются, граф растет [0.2, 256]

В неориентированный граф последовательно добавляются новые ребра. Изначально граф пустой. После каждого добавления нужно говорить, является ли текущий граф двудольным.

Формат входных данных

На первой строке n — количество вершин, m — количество операций «добавить ребро». Следующие m строк содержат пары чисел от 1 до n — описание добавляемых ребер.

Формат выходных данных

Выведите в строчку m нулей и единиц. i-й символ должен быть равен единице, если граф, состоящий из первых i ребер, является двудольным.

Система оценки

Подзадача 1 (25 баллов) $1 \leqslant n, m \leqslant 1\,000$.
Подзадача 2 (50 баллов) $1 \leqslant n, m \leqslant 50\,000$.
Подзадача 3 (25 баллов) $1 \leqslant n, m \leqslant 300\,000$.

Примеры

stdin	stdout
3 3 1 2 2 3 3 1	110

Задача 21F. Электросеть [0.1, 256]

Дан граф. Вершины – точки на плоскости. Между каждой парой городов i, j можно построить дорогу. Стоимость дороги $(|x_i - x_j| + |y_i - y_j|) \cdot (k_i + k_j)$. А ещё в каждом городе можно построить электростанцию за стоимость c_i. Изначально дорог нет. Нужно за минимальную суммарную стоимость сделать так, чтобы в каждой компоненте связности была минимум одна электростанция.

Формат входных данных

В первой строке записано одно целое число n $(1 \leqslant n \leqslant 2000)$ — количество городов. Затем следует n строк. В i-й строке записаны два целых числа x_i $(1 \leqslant x_i \leqslant 10^6)$ и y_i $(1 \leqslant y_i \leqslant 10^6)$ — координаты i-го города. В следующей строке записаны n целых чисел $c_1, c_2, \ldots, c_n$ $(1 \leqslant c_i \leqslant 10^9)$ — цена установки электростанции в i-м городе. В последней строке записаны n целых чисел $k_1, k_2, \ldots, k_n$ $(1 \leqslant k_i \leqslant 10^9)$.

Формат выходных данных

В первой строке выведите суммарную стоимость.

Далее информацию где строить электростанции – число городов, сами города.

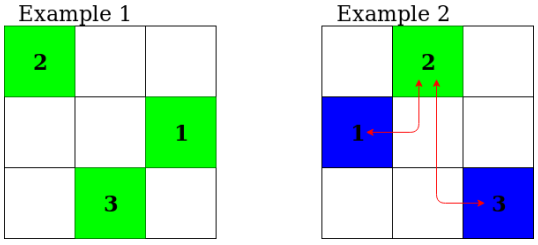
Далее информацию, какие города соединять – число дорог, какие города соединять.

Если существует несколько способов так выбрать города и соединения, чтобы получить конструкцию минимальной цены, то выведите любую из них.

Примеры

stdin	stdout
3 2 3 1 1 3 2 3 2 3 3 2 3	8 3 1 2 3 0
3 2 1 1 2 3 3 23 2 23 3 2 3	27 1 2 2 1 2 2 3

Замечание



Города с электростанциями раскрашены зеленым, остальные — синим, дороги красным.

В первом примере цена строительства электростанций во всех городах равна $3+2+3 = 8$. Можно показать, что больше никакая конфигурация не стоит меньше 8 иен.

Во втором примере цена строительства электростанции в городе 2 равна 2. Стоимость соединения городов 1 и 2 равна $2 \cdot (3+2)$, соединения городов 2 и 3 равна $3 \cdot (2+3)$. Итого 27.

Для искателей острых ощущений

Задача 21G. Возьми себе за правило: летай всегда GraphAero! [0.2, 256]

Наконец авиаперевозки стали доступны всем и каждому! Однако, из-за жёсткой конкуренции в сфере пассажироперевозок осталось только две авиакомпании: «GraphAero Airlines» и «Aerofloat».

Авиакомпания «GraphAero Airlines» активно развивается. Ведь для получения большей прибыли... простите, для удобства пассажиров каждый месяц компания добавляет один новый рейс. Компании «Aerofloat» остаётся довольствоваться тем, что остаётся. А именно, единственная возможность удержаться на плаву — добавлять рейсы, дублирующие самые загруженные рейсы компании «GraphAero Airlines». Рейс является самым загруженным, если существует такая пара городов, что можно долететь (возможно, с пересадками) из одного города в другой, используя рейсы авиакомпании, но если этот рейс отменить — то долететь будет невозможно. Аналитикам компании «Aerofloat» необходимо постоянно контролировать ситуацию — сколько в данный момент существует самых загруженных рейсов.

Поскольку вы уже давно мечтаете летать по льготным ценам (скидка $10^{-5}\%$), вы решили оказать посильную помощь. Помните: самолёты летают по всему миру! Между двумя крупными городами может быть более одного рейса, а города бывают настолько большими, что самолёты могут летать в пределах одного города. Рейсами можно пользоваться как в одну, так и в другую сторону.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 100\,000$) — количество городов и M ($0 \leqslant M \leqslant 100\,000$) — изначальное число рейсов компании «GraphAero Airlines». Далее следует M строк, в каждой содержится описание очередного рейса — номера двух городов, между которыми осуществляется рейс. В следующей строке содержится число K ($1 \leqslant K \leqslant 100\,000$) — количество добавленных рейсов. Далее содержится описание добавленных рейсов в таком же формате.

Формат выходных данных

После каждого добавления нового рейса выведите на отдельной строке одно число — количество самых загруженных рейсов.

Примеры

stdin	stdout
4 0 4 1 2 2 3 3 4 1 4	1 2 3 0
4 3 1 2 2 3 3 4 4 1 1 1 2 1 3 1 4	3 2 1 0

Задача 21H. Два китайца и дерево [0.3, 256]

Дан ориентированный взвешенный граф. Выбрать множество рёбер минимально возможного размера и при равенстве размера минимального суммарного веса, чтобы все вершины были достижимы из **первой**. Гарантируется, что такое множество существует.

Формат входных данных

Входные данные содержат описание одного или более тестов.

Каждый тест описывается следующим образом. На первой строке количества вершин и рёбер в графе n ($1 \leqslant n \leqslant 1\,000$) и m ($1 \leqslant m \leqslant 3\,000$). На следующих m строках рёбра в формате «a_i b_i w_i» ($1 \leqslant a_i, b_i \leqslant m$, $-10^5 < w_i < 10^5$), что обозначает ребро из вершины a_i в вершину b_i веса w_i. В графе могут быть и петли, и кратные рёбра.

Сумма n по всем тестам не более $1\,000$. Сумма m по всем тестам не более $3\,000$.

Формат выходных данных

Для каждого теста выведите одно число — суммарный вес выбранных рёбер.

Примеры

stdin	stdout
3 3 1 2 -3 2 3 -1 3 1 -10 4 5 1 2 10 2 4 2 1 3 10 3 4 1 1 4 10 5 6 1 2 1 2 3 1 3 4 1 4 5 1 1 5 2 5 3 -1	-4 21 3

Задача 21I. MST случайных точек [1.0, 256]

Даны n различных точек на плоскости. Координаты точек — целые числа от 0 до 30 000 включительно. Точки выбраны *случайно* в следующем смысле: рассмотрим все возможные наборы из n различных точек на плоскости с заданными ограничениями на координаты и выберем из них случайно и равновероятно один набор.

Вы можете провести отрезок между любыми двумя заданными точками. Длина отрезка между точками с координатами (x_1, y_1) и (x_2, y_2) равна $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Будем говорить, что точки a и b *связаны*, если они соединены отрезком, или же существует точка d, которая связана и с a, и с b. Ваша задача — провести отрезки минимальной суммарной длины так, чтобы все точки были связаны.

Формат входных данных

В первой строке ввода задано целое число n ($2 \leqslant n \leqslant 50\,000$). Следующие n строк содержат координаты точек. Гарантируется, что все точки различны. Кроме того, во всех тестах, кроме примера, гарантируется, что точки выбраны случайно, как описано в условии.

Формат выходных данных

В первой строке выведите вещественное число w — суммарную длину отрезков. В следующих $(n - 1)$ строках выведите отрезки, по одному на строке. Каждый отрезок следует выводить как два числа от 1 до n, обозначающие номера точек, являющихся концами этого отрезка.

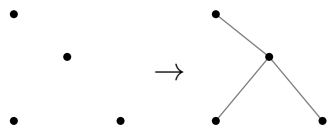
Пусть на самом деле суммарная длина выведенных вами отрезков равна w^*, а суммарная длина отрезков в оптимальном ответе равна w_{opt}. Тогда ваш ответ будет считаться верным, если

$$\max\left(\left|\frac{w}{w^*} - 1\right|, \left|\frac{w^*}{w_{\text{opt}}} - 1\right|\right) < 10^{-12}.$$

Пример

stdin	stdout
4 0 10 5 6 10 0 0 0	22.02362358924615 1 2 2 3 4 2

Иллюстрация



Задача 21J. Таможенные пошлины [0.2, 256]

Недавно королева страны AlgoLand придумала новый способ отмывания денег для своего королевского двора. Она решила, что всякий житель, желающий совершить путешествие из одного города страны в другой, должен расплатиться за это желание своими деньгами.

В стране AlgoLand есть N городов, пронумерованных от 1 до N. Некоторые города соединены дорогами, движение по которым разрешено в двух направлениях. Начиная движение по какой-нибудь дороге, путешественник обязательно должен доехать до ее конца.

Предположим теперь, что житель страны хочет совершить путешествие из города A в город B. Новый указ королевы гласит, что при проезде по любой дороге страны во время этого путешествия, полицейские могут взять с этого жителя таможенную пошлину в пользу королевского двора (а могут и не взять). Если при этом у жителя недостаточно денег для уплаты пошлины, то он автоматически попадает в тюрьму. Указ также устанавливает величину пошлины для каждой дороги страны. Так как королева заботится о жителях своей страны, то она запретила полицейским брать с жителя пошлину более чем три раза во время одного путешествия.

Отметим, что если существует несколько способов попасть из города A в город B, то житель может выбрать для путешествия любой из них по собственному желанию.

Напишите программу, которая определяет, какую минимальную сумму денег должен взять с собой житель, чтобы гарантированно не попасть в тюрьму во время путешествия.

Формат входных данных

Первая строка входного файла содержит числа N и M ($2 \leqslant N \leqslant 10\,000$, $1 \leqslant M \leqslant 100\,000$), разделенные пробелом — количества городов и дорог. Следующие M строк описывают дороги. Каждая из этих строк описывает одну дорогу и содержит три числа X, Y, Z ($1 \leqslant X, Y \leqslant N$; $X \neq Y$; $1 \leqslant Z \leqslant 1\,000\,000\,000$), разделенных пробелами, означающие, что дорога соединяет города X и Y и пошлина за проезд по ней равна Z денежных единиц. Все числа Z целые. Последняя строка содержит числа A и B ($1 \leqslant A, B \leqslant N$; $A \neq B$) — номера начального и конечного городов путешествия. Гарантируется, что существует хотя бы один способ проезда из A в B.

Формат выходных данных

Единственная строка выходного файла должна содержать одно число, равное минимальной сумме денег, которую должен взять с собой житель, чтобы иметь возможность совершить путешествие из города A в город B и при этом гарантированно не попасть в тюрьму независимо от действий полицейских.

Пример

stdin	stdout
5 6 1 2 10 1 3 4 3 2 3 1 4 1 4 5 2 5 2 3 1 2	6