Задача A. Кратчайший путь

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

Дан ориентированный взвешенный граф. Найдите кратчайшее расстояние от одной заданной вершины до другой.

Формат входных данных

В первой строке входного файла три числа: N, S и F ($1 \leqslant N \leqslant 2000, 1 \leqslant S, F \leqslant N$), где N — количество вершин графа, S — начальная вершина, а F — конечная. В следующих N строках по N чисел — матрица смежности графа, где -1 означает отсутствие ребра между вершинами, а любое неотрицательное число — присутствие ребра данного веса. Вес каждого ребра не превышает 10^9. На главной диагонали матрицы всегда нули.

Формат выходных данных

Вывести искомое расстояние или -1, если пути между указанными вершинами не существует.

Пример

pathmgep.in	pathmgep.out
3 1 2 0 -1 2 3 0 -1 -1 4 0	6

Задача B. Кратчайший путь-2

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

Дан неориентированный связный взвешенный граф. Найдите кратчайшее расстояние от первой вершины до всех вершин.

Формат входных данных

В первой строке входного файла два числа: n и m ($2 \leqslant n \leqslant 30000, 1 \leqslant m \leqslant 400000$), где n — количество вершин графа, а m — количество ребер.

Следующие m строк содержат описание ребер. Каждое ребро задается стартовой вершиной, конечной вершиной и весом ребра. Вес каждого ребра — неотрицательное целое число, не превосходящее 10^4.

Формат выходных данных

Выведите n чисел — для каждой вершины кратчайшее расстояние до нее.

Пример

pathbgep.in	pathbgep.out
4 5 1 2 1 1 3 5 2 4 8 3 4 1 2 3 3	0 1 4 5

Задача C. Флойд

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

Полный ориентированный взвешенный граф задан матрицей смежности. Постройте матрицу кратчайших путей между его вершинами. Гарантируется, что в графе нет циклов отрицательного веса.

Формат входных данных

В первой строке вводится единственное число N ($1 \leqslant N \leqslant 100$) — количество вершин графа. В следующих N строках по N чисел задается матрица смежности графа (j-ое число в i-ой строке — вес ребра из вершины i в вершину j). Все числа по модулю не превышают 100. На главной диагонали матрицы — всегда нули.

Формат выходных данных

Выведите N строк по N чисел — матрицу расстояний между парами вершин, где j-ое число в i-ой строке равно весу кратчайшего пути из вершины i в j.

Пример

floyd.in	floyd.out
4 0 5 9 100 100 0 2 8 100 100 0 7 4 100 100 0	0 5 7 13 12 0 2 8 11 16 0 7 4 9 11 0

Задача D. Цикл отрицательного веса

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

Дан ориентированный граф. Определите, есть ли в нем цикл отрицательного веса, и если да, то выведите его.

Формат входных данных

Во входном файле в первой строке число N ($1 \leqslant N \leqslant 100$) — количество вершин графа. В следующих N строках находится по N чисел — матрица смежности графа. Все веса ребер не превышают по модулю 10 000. Если ребра нет, то соответствующее число равно 100 000.

Формат выходных данных

В первой строке выходного файла выведите «`YES`», если цикл существует или «`NO`» в противном случае. При его наличии выведите во второй строке количество вершин в искомом цикле и в третьей строке — вершины входящие в этот цикл в порядке обхода.

Пример

negcycle.in	negcycle.out
2 0 -1 -1 0	YES 2 2 1

Задача E. Кратчайшие пути

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

Вам дан взвешенный ориентированный граф и вершина s в нём. Для каждой вершины графа u выведите длину кратчайшего пути от вершины s до вершины u.

Формат входных данных

Первая строка входного файла содержит три целых числа n, m, s — количество вершин и ребёр в графе и номер начальной вершины соответственно ($2 \leqslant n \leqslant 2\,000$, $1 \leqslant m \leqslant 5\,000$).

Следующие m строчек описывают рёбра графа. Каждое ребро задаётся тремя числами — начальной вершиной, конечной вершиной и весом ребра соответственно. Вес ребра — целое число, не превосходящее 10^{15} по абсолютной величине. В графе могут быть кратные рёбра и петли.

Формат выходных данных

Выведите n строчек — для каждой вершины u выведите длину кратчайшего пути из s в u. Если не существует пути между s и u, выведите «*». Если не существует кратчайшего пути между s и u, выведите «-».

Пример

path.in	path.out
6 7 1 1 2 10 2 3 5 1 3 100 3 5 7 5 4 10 4 3 -18 6 1 -1	0 10 - - - *

Задача F. Кратчайший путь длины K

Имя входного файла:	`kpath.in`
Имя выходного файла:	`kpath.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный граф. Найдите кратчайшие пути, состоящие из K рёбер, от S до всех вершин.

Формат входных данных

В первой строке дано целых четыре целых числа: $1 \leqslant N, M \leqslant 10^4$ — количества вершин и рёбер, $0 \leqslant K \leqslant 100$ — количество рёбер в кратчайших путях, $1 \leqslant S \leqslant N$ — начальная вершина.

В последующих M строках даны тройки целых чисел a_i, b_i, w — начало и конец ребра, а также его вес ($1 \leqslant a_i, b_i \leqslant N, -10^5 \leqslant w \leqslant 10^5$).

Формат выходных данных

Выведите ровно N чисел по одному в строке. i-е число — длина минимального пути из ровно K рёбер из S в i, или -1, если пути не существует.

Примеры

kpath.in	kpath.out
3 3 1 1 1 2 100 2 3 300 1 3 2	-1 100 2
3 3 2 1 1 2 100 2 3 300 1 3 2	-1 -1 400

Задача G. Roadblock

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

Каждое утро Фермер Джон идет от дома к амбару. Ферма представляет собой множество из n полей (дом на поле 1, амбар на поле n), соединенных m двунаправленными дорогами, с каждой из которых ассоциирована длина.

Никакие два поля не соединены более чем одной дорогой, и существует маршрут дорог от любого поля к любому. Когда ФД идет от одного поля к другому, он всегда выбирает маршрут, состоящий из последовательности дорог, которые дают минимальную суммарную длину.

Коровы решили сделать ФД маленькую неприятность, выложив сено на одной из m дорог, тем самым удваивая ее длину.

Коровы хотят выбрать такую дорожку, чтобы максимально увеличить расстояние, которое ФД пройдет от дома до амбара. Помогите коровам определить, насколько они удлинят маршрут ФД.

Формат входных данных

Первая строка содержит два целых числа n и m ($1 \leqslant n \leqslant 100$; $1 \leqslant m \leqslant 10\,000$).

Следующие m строк описывают дороги. i-я из этих строк состоит из трех целых чисел u_i, v_i и l_i — дорога длины l_i соединяет поля с номерами v_i и u_i ($1 \leqslant u_i, v_i \leqslant n$; $1 \leqslant l_i \leqslant 10^6$).

Формат выходных данных

Выведите максимально возможное увеличение общей длины кратчайшего маршрута, которого можно добиться удвоением длины одной дороги.

Примеры

rblock.in	rblock.out
5 7 2 1 5 1 3 1 3 2 8 3 5 7 3 4 3 2 4 7 4 5 2	2

Замечание

Если коровы удвоят длину дороги от поля 3 к полю 4 (от 3 до 6), тогда кратчайшим маршрутом станет путь 1-3-5, с общей длиной $1 + 7 = 8$. Что на 2 больше, чем исходный кратчайший маршрут.

Задача H. Кратчайший путь

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

Дан ориентированный взвешенный ациклический граф. Требуется найти в нем кратчайший путь из вершины s в вершину t.

Формат входных данных

Первая строка входного файла содержит четыре целых числа n, m, s и t — количество вершин, дуг графа, начальная и конечная вершина соответственно. Следующие m строк содержат описания дуг по одной на строке. Ребро номер i описывается тремя целыми числами b_i, e_i и w_i — началом, концом и длиной дуги соответственно ($1 \leqslant b_i, e_i \leqslant n$, $|w_i| \leqslant 1000$).

Входной граф не содержит циклов и петель.

$1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 200\,000$.

Формат выходных данных

Первая строка выходного файла должна содержать одно целое число — длину кратчайшего пути из s в t. Если пути из s в t не существует, выведите «Unreachable».

Примеры

dag-shortpath.in	dag-shortpath.out
2 1 1 2 1 2 -10	-10
2 1 2 1 1 2 -10	Unreachable

Задача I. Транспортировка

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

Компания “Яндекс” решила подарить всем школьникам и преподавателям ЛКШ оригинальные кру́жки. К сожалению, количество необходимых кружек оказалось столь велико, что изготовитель доставил кружки в офис Яндекса в самый последний момент. До открытия смены в "Берендеевых полянах"осталось всего 24 часа.

О плачевном состоянии дорог по пути на базу ходят легенды — в частности, на многих разбитых дорогах действует ограничение на вес автомобиля. Соответственно, от нагруженности машины зависит возможность воспользоваться тем или иным маршрутом, тяжёлой машине может потребоваться ехать в обход.

Уже совершенно очевидно, что все кружки не поспеют к открытию. Чтобы спасти ситуацию, отвезите первым рейсом максимально возможное количество кружек успев до начала открытия смены.

Формат входных данных

В первой строке находятся целые числа n ($2 \leqslant n \leqslant 500$) и m — количество городов и количество двусторонних дорог, соответственно.

В следующих m строках описываются дороги.

В каждой строке находятся целые числа a_i, b_i, t_i, w_i — соответственно два города, ею соединяемые, время на проезд по ней в минутах и ограничение на вес автомобиля в граммах ($t_i \leqslant 1440$, $w_i \leqslant 10^9$, $1 \leqslant a_i, b_i \leqslant n$)

Между каждой парой городов есть не более одной дороги.

Кроме того, известно, что офис Яндекса имеет номер 1, а “Берендеевы поляны” — номер n, одна кружка весит 100 грамм, а пустой грузовик — 3 тонны.

Формат выходных данных

Выведите одно число — максимальное количество кружек, которое можно привезти, потратив не более 24 часов.

Пример

cups.in	cups.out
3 3 1 2 10 3000220 2 3 20 3000201 1 3 1 3000099	2

Задача J. На санях

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

В начале XIX века еще не было самолетов, поездов и автомобилей, поэтому все междугородние зимние поездки совершались на санях. Как известно, с дорогами в России тогда было даже больше проблем, чем сейчас, а именно на n существовавших тогда городов имелась ровно $n-1$ дорога, каждая из которых соединяла ровно два города. К счастью, из каждого города можно было добраться в любой другой (возможно, через некоторые промежуточные города). В каждом городе имелась почтовая станция (или, как ее называют, «ям»), на которой можно было пересесть в другие сани. При этом ямщики могли долго запрягать (для каждого из городов известно время, которое ямщики в этом городе тратят на подготовку саней к поездке) и быстро ехать (также для каждого города известна скорость, с которой ездят ямщики из него). Можно считать, что количество ямщиков в каждом городе не ограничено.

Если бы олимпиада проводилась 200 лет назад, то путь участников занимал бы гораздо большее время, чем сейчас. Допустим, из каждого города в Москву выезжает участник олимпиады и хочет добраться до Москвы за наименьшее время (не обязательно по кратчайшему пути: он может заезжать в любые города, через один и тот же город можно проезжать несколько раз). Сначала он едет на ямщике своего города. Приехав в любой город, он может либо сразу ехать дальше, либо пересесть. В первом случае он едет с той же скоростью, с какой ехал раньше. Решив сменить ямщика, он сначала ждет, пока ямщик подготовит сани, и только потом едет с ним (естественно, с той скоростью, с которой ездит этот ямщик). В пути можно делать сколько угодно пересадок.

Жюри стало интересно, какое время необходимо, чтобы все участники олимпиады доехали из своего города в Москву 200 лет назад. Все участники выезжают из своих городов одновременно.

Формат входных данных

В первой строке входного файла дано натуральное число n, не превышающее 2000 — количество городов, соединенных дорогами. Город с номером 1 является столицей. Следующие n строк содержат по два целых числа: t_i и v_i — время подготовки саней в городе i, выраженное в часах, и скорость, с которой ездят ямщики из города i, в километрах в час ($0 \leqslant t_i \leqslant 100, 1 \leqslant v_i \leqslant 100$). Следующие $n-1$ строк содержат описания дорог того времени. Каждое описание состоит из трех чисел a_j, b_j и s_j, где a_j и b_j — номера соединенных городов, а s_j — расстояние между ними в километрах ($1 \leqslant a_j \leqslant n, 1 \leqslant b_j \leqslant n, a_j \neq b_j, 1 \leqslant s_j \leqslant 10000$). Все дороги двусторонние, то есть если из a можно проехать в b, то из b можно проехать в a. Гарантируется, что из всех городов можно добраться в столицу.

Формат выходных данных

Сначала выведите одно вещественное число — время в часах, в которое в Москву приедет последний участник. Далее выведите путь участника, который приедет самым последним (если таких участников несколько, выведите путь любого из них). Выведите город, из которого этот участник выехал первоначально, и перечислите в порядке посещения те города, в которых он делал пересадки. Последовательность должна заканчиваться столицей. При проверке ответ будет засчитан, если из трех величин «время путешествия по выведенному пути», «выведенное время» и «правильный ответ» каждые две отличаются менее чем на 0.0001.

Система оценки

Номер подзадачи	Баллы	Ограничения	Комментарии
		n	
0	0		Примеры из условия.
1	28	$1 \leqslant n \leqslant 40$	Баллы начисляются, если все тесты пройдены.
2	35	$1 \leqslant n \leqslant 400$	Баллы начисляются, если все тесты этой и предыдущих групп пройдены.
3	37	$1 \leqslant n \leqslant 2000$	Баллы начисляются, если все тесты этой и предыдущих групп пройдены.

Примеры

стандартный ввод	стандартный вывод
4 1 1 10 30 5 40 1 10 1 2 300 1 3 400 2 4 100	31.000000000000000 4 2 1
3 1 1 0 10 0 55 1 2 100 2 3 10	3.000000000000000 2 3 1