Задача 1. Шлю я за пакетом пакет

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

В базе данных роутера хранится информация о нескольких серверах, некоторые из них связаны между собой напрямую, другие — только опосредованно. Получая электронное письмо от сервера с номером M (отправителя), предназначенное для сервера с номером N (получателя), роутер должен найти в своей базе данных самый короткий путь пересылки этого письма по сети.

Напишите программу, которая, зная информацию о всевозможных соединениях и их длительности, осуществляла бы поиск самого короткого пути пересылки и выдавала бы информацию о времени, за которое письмо преодолеет весь этот путь.

Формат входных данных

В первой строке входного файла содержится одно натуральное число N — количество серверов, информация о которых записана в базе данных роутера ($1 \leqslant N \leqslant 100$).

Во второй строке — два целых числа S_1 и S_2, разделенные пробелом — номера сервера-отправителя и сервера-получателя ($1 \leqslant S_1, S_2 \leqslant N$).

Начиная с третьей строки и до конца файла, записаны имеющиеся между серверами активные каналы связи и скорость передачи данных по этим каналам. Сначала записаны номера серверов S_i и S_j, а затем скорость передачи данных между ними — целое число K ($1 \leqslant S_i, S_j \leqslant N, 0 \leqslant K \leqslant 1000$). Все числа находятся на одной строке и разделены пробелами.

Формат выходных данных

В выходной файл нужно записать одно целое число – время, необходимое письму для прохождения по самому быстрому пути связи.

Если такого пути нет (например, на промежуточном сервере произошла авария), то необходимо выдать сообщение `no`.

Примеры

input.txt	output.txt
4 4 1 1 2 10 1 3 2 2 4 1 3 4 10	11
4 1 4 1 2 100 1 3 20 2 3 57	no

Задача 2. Фиолетовое такси

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

«Фиолетовое такси» предложило клиентам новую услугу — теперь можно узнать кратчайшее время проезда для любых мест отправления и назначения. Однако диспетчеры не успевают отвечать на все запросы. Напишите программу, которая поможет диспетчерам быстро находить минимальное время пути. Учтите, что в редких случаях клиента интересует не только время, но и подробное описание предполагаемого маршрута.

Формат входных данных

В первой строке входного файла записано четыре целых числа: N — количество пунктов, в которых люди садятся и выходят из такси, M — количество дорог, их соединяющих, P — количество запросов на поиск кратчайшего пути, K — количество запросов на поиск минимального времени ($1 \leqslant N, P \leqslant 300$, $1 \leqslant M, K \leqslant 50\,000$).

Далее в M строках описываются дороги, по три целых числа в каждой строке — номера двух пунктов, соединенных этой дорогой и время L_i проезда по дороге ($1 \leqslant L_i \leqslant 10^6$).

Пункты пронумерованы числами от 1 до N. Все дороги двусторонние. Пару пунктов **может** соединять несколько дорог. Гарантируется, что все пункты соединены между собой.

Далее следуют запросы, по одному запросу в строке. Сначала записаны запросы на поиск кратчайшего пути (P штук), затем запросы на поиск минимального времени (K штук). Каждый запрос описан двумя целыми числами S_j и T_j — номер пункта отправления и номер пункта назначения соответственно ($1 \leqslant S_j \neq T_j \leqslant N$).

Формат выходных данных

Для каждого запроса нужно вывести ответ на отдельной строке. Во-первых, нужно вывести минимальное время пути от пункта отправления до пункта назначения. Во-вторых, для запроса на поиск кратчайшего пути нужно вывести дополнительно $(A_j + 1)$ целых чисел — описание оптимального маршрута. Первое из этих чисел должно быть равно A_j — количеству вершин в пути (включая пункты отправления и назначения), а остальные числа должны задавать номера пунктов, по которым идёт маршрут, в порядке их прохождения.

Пример

input.txt	output.txt
5 6 3 2 4 2 2 1 4 8 2 3 6 1 5 7 2 1 3 4 3 9 3 5 1 3 3 1 4 3 5 4	16 4 3 2 1 5 9 3 1 2 3 9 3 3 2 1 8 12

Задача 3. Транзитивное замыкание

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	разумное

Дано бинарное отношение R над множеством чисел $X = \{1, 2, 3, \ldots, N\}$. Требуется найти его рефлексивно-транзитивное замыкание.

Указание: Используйте алгоритм Уоршалла (Флойда-Уоршалла).

Формат входных данных

В первой строке входного файла записано одно целое число N — размер множества ($1 \leqslant N \leqslant 500$).

Далее идёт N строк по N символов в каждой, задающие отношение R. j-ый символ i-ой строки равен 1, если пара $(i, j) \in R$ (т.е. лежит в отношении R), и равен 0 в противном случае.

Формат выходных данных

В выходной файл необходимо вывести N строк по N символов в каждой — рефлексивное и транзитивное замыкание отношения R, описанное в том же формате, что и исходное отношение R во входных данных.

Пример

input.txt	output.txt
5 00001 10010 01101 10000 00011	10011 11011 11111 10011 10011

Задача 4. COVID-19: Карантин

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вовочка грустит: из-за пандемии коронавируса его город закрыли на «самоизоляцию», и его лично — на 14-дневный карантин. Теперь он не может выходить из дома без необходимости. Если он попробует, то его поймают суровые дядечки и упекут в инфекционный стационар. А Вовочке очень хочется встретиться с друзьями и рассказать им о своей заграничной поездке! Будучи от природы человеком безответственным и безрассудным, Вовочка нашёл выход из положения.

Оказывается, выгуливать собак никто не запрещает даже тем, кто на карантине! К счастью, у Вовочки есть крупная и хорошо дрессированная собака по кличке Слоняша, с которой он может свободно гулять. Вот только до некоторых друзей идти далеко, а опыт показывает, что если Слоняша долго не делает того, что обычно должны делать выгуливаемые собаки, то наблюдатели начинают подозревать неладное и быстро отправляют Вовочку домой. Таким образом, если Вовочка хочет пойти к кому-то в гости, ему надо составить такой маршрут, чтобы на протяжении всего пути Слоняша была непрерывно занята делом. А чтобы пополнять запасы драгоценной жидкости, Слоняша может пить воду из луж — благо на дворе весна и луж на дорогах предостаточно.

Вовочка решил прибегнуть к помощи вычислительной техники для составления плана. Он смоделировал проблему следующим образом. Есть ориентированный граф, рёбра которого представляют собой дорожки на улице (да, его родной город настолько суров, что на всех пешеходных дорожках одностороннее движение). Каждому ребру приписан вес, который определяет, сколько миллилитров жидкости нужно Слоняше на его качественное преодоление. Вес ребра рассчитывается за вычетом той воды, которую можно выпить из расположенных на дорожке луж, так что он может быть отрицательным.

Вовочка хочет найти оптимальный путь от своего дома до дома каждого из своих друзей. Пусть считается оптимальным, если запасы жидкости у Слоняши в конце пути максимально возможные, а значит суммарный вес всех дорожек в пути минимально возможен. Вовочка надеется, что если умело расходовать драгоценные миллилитры, то можно будет потом сходить куда-нибудь ещё!

Помогите Вовочке решить его нечеловеческую проблему.

Формат входных данных

В первой строке входного файла записано три целых числа: N — количество вершин в графе, M — количество дорожек, K — количество друзей у Вовочки ($2 \leqslant N \leqslant 5\,000$, $1 \leqslant M \leqslant 50\,000$, $1 \leqslant K \leqslant 50$).

Во второй строке записано K различных целых чисел: номера вершин v_j, в которых живут друзья Вовочки ($2 \leqslant v_j \leqslant N$). Вовочка живёт в вершине номер один.

Далее в M строках описываются дорожки, по три целых числа в каждой строке: a_i — вершина, в которой начинается дорожка, b_i — вершина, в которой заканчивается дорожка и w_i — вес дорожки в миллилитрах ($1 \leqslant a_i, b_i \leqslant N$, $|w_i| \leqslant 10^5$).

Гарантируется, что в графе нет циклов отрицательного веса. Видимо, идея с выгулом

собаки Вовочке не первому пришла в голову, и все подобные циклы уже тщательно выпиты.

Формат выходных данных

Для каждого из K друзей Вовочки нужно вывести оптимальный путь до его дома, в порядке задания этих домов во входных данных. Каждый путь описывается в отдельной строке. Сначала должен быть записан вес пути, который должен быть минимально возможным, затем количество вершин в пути, и наконец номера всех этих вершин в порядке прохождения по пути.

Гарантируется, что до каждого друга можно добраться по дорожкам.

Пример

input.txt	output.txt
5 9 2 2 4 1 3 1000 3 2 300 1 2 1200 1 4 500 4 5 400 5 4 0 4 3 600 3 5 -300 5 2 200	900 4 1 3 5 2 500 2 1 4

Комментарий

Учтите, что автор условия **не** разделяет легкомысленное отношение Вовочки к карантину и к жизни вообще.

Задача 5. Сплетницы

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	3 секунды*
Ограничение по памяти:	256 мегабайт

Домохозяйки проводят много времени за телефонными разговорами с подругами. Как только одной из них становится что-нибудь известно, она садится за телефон и сообщает свежую сплетню подругам. С каждой из своих подруг домохозяйка заканчивает разговор через разное время. Напишите программу, вычисляющую минимальное время, за которое сплетня дойдет от домохозяйки-распространительницы свежей сплетни, живущей в дома S, до другой, живущей в доме T.

Формат входных данных

В первой строке файла содержатся целые числа N, M, K — количество домохозяек, количество пар любящих поговорить домохозяек, и количество запросов ($1 \leqslant N \leqslant 3\,000$, $1 \leqslant M \leqslant 300\,000$, $1 \leqslant K \leqslant 20$).

Далее идут K строк с запросами. В каждой строке два целых числа S_i и T_i — номер дома, где зарождается сплетня, и номер дома, в которую она попадёт ($1 \leqslant S_i \neq T_i \leqslant N$).

В каждой из следующих строк располагаются по три целых числа, из которых первые два числа — это номера домов домохозяек, которые любят разговаривать по телефону друг с другом, третье число — время разговора между ними — неотрицательно и не превосходит 100 000.

Формат выходных данных

Для каждого запроса во входных данных нужно вывести ответ на задачу в отдельной строке.

Если сплетня не может дойти до адресата, то выведите лишь слово "`NO`".

В противном случае выведите слово "`YES`", затем искомое минимальное время C и количество сплетниц P в критическом пути, на котором это время достигается. Наконец, выведите P целых чисел — номера домов на критическом пути в том порядке, в котором по ним будет проходить сплетня.

Пример

input.txt	output.txt
7 6 4 1 7 7 1 5 3 4 6 1 2 2 1 3 1 3 6 1 3 7 2 6 7 1 2 4 1	YES 3 3 1 3 7 YES 3 4 7 6 3 1 NO YES 5 5 4 2 1 3 6

Задача 6. Спички

Источник:	Повышенной сложности
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Во время игры какое-то количество спичек рассыпается по столу и игроки должны убрать их, выбирая по одной, не сдвинув с места остальные спички.

Вам нужно определить, касаются ли данные спички друг друга. Вам будет дан список координат концов спичек, и нужно будет определить, соединены ли две данные спички или нет. Заметим, что касание – это соединение, и что две спички могут быть соединены, если они соединены не прямо, а посредством других спичек.

Формат входных данных

Входной файл сдержит несколько тестов.

В первой строке каждого теста находится целое число n , задающее количество спичек на столе ($1 < n < 13$).

Каждая из следующих n строк содержит 4 положительных целых x_1, y_1, x_2, y_2, обозначающих координаты (x_1, y_1), (x_2, y_2) концов одной спички. Все координаты меньше 100. Заметим, что спички могут быть разной длины. Первая введенная спичка будет спичкой номер 1, вторая, соответственно, номер 2 и т.д. Все данные корректны, и нет спичек нулевой длины.

Остальные строки теста содержат по два целых числа a и b (от 1 до n, включительно). Вам нужно определить, связана ли спичка a со спичкой b, или нет.

Конец теста: $a = b = 0$. После каждого теста во входном файле пустая строка.

Последняя строка в файле содержит один символ — 0.

Формат выходных данных

Вам нужно для каждой пары спичек сгенерировать строчку `CONNECTED`, если данные спички связаны, и `NOT CONNECTED`, наоборот. Будем считать, что спичка связана сама с собой.

Пустые строки между ответами не вставлять.

Пример

input.txt	output.txt
7 1 6 3 3 4 6 4 9 4 5 6 7 1 4 3 5 3 5 5 5 5 2 6 3 5 4 7 2 1 4 1 6 3 3 6 7 2 3 1 3 0 0 0	CONNECTED NOT CONNECTED CONNECTED CONNECTED NOT CONNECTED CONNECTED