

Содержание

Задачи	2
Задача А. Стоимость проезда [0.3 sec, 256 mb]	2
Задача В. Сумма расстояний [0.3 sec, 256 mb]	3
Задача С. Кратчайший путь двух коней [0.3 sec, 256 mb]	4
Задача D. Island. Островные государства [0.5 sec, 256 mb]	5
Задача Е. Диаметр графа [0.3 sec, 256 mb]	6
Задача F. Расстояние между вершинами [1 sec, 256 mb]	7
Задача G. Расстояние между вершинами [1 sec, 256 mb]	8
Задача H. Path. Кратчайший путь [1 sec, 256 mb]	9
Бонус	10
Задача I. Кратчайший путь [2 sec, 256 mb]	10
Задача J. Экскурсия [0.2 sec, 256 mb]	11
Задача K. Грязь [0.15 sec, 256 mb]	12
Задача L. Таможенные пошлины [1 sec, 256 mb]	13

В некоторых задачах большой ввод и вывод. Имеет смысл пользоваться супер быстрым вводом-выводом: <http://acm.math.spbu.ru/~sk1/algo/input-output/>

Задачи

Задача А. Стоимость проезда [0.3 сек, 256 mb]

Страна состоит из n городов и m дорог. Города пронумерованы числами от 1 до n . Город с номером s является столицей. Все дороги односторонние, проход по каждой дороге стоит ровно 1 золотой. Требуется найти минимальные стоимости проезда от каждого города до столицы.

Формат входных данных

В первой строке файла записаны три целых числа — n , s и m (количество городов, номер столичного города и количество дорог).

В следующих m строках записаны пары чисел. Пара чисел (a, b) означает, что есть дорога из города a в город b .

Ограничения: $1 \leq n \leq 10^5, 0 \leq m \leq 10^5$.

Формат выходных данных

Выведите n чисел — минимальные стоимости проезда от городов до столицы. Если от какого-то города не существует ни одного пути до столицы, выведите -1 .

Пример

bfsrev.in	bfsrev.out
3 2 2	1 0 -1
1 2	
2 3	

Задача В. Сумма расстояний [0.3 sec, 256 mb]

Дан связный неориентированный граф. Требуется найти сумму расстояний между всеми парами вершин.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($1 \leq n \leq 1000$, $0 \leq m \leq 10\,000$).

Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — номерами концов ребра ($1 \leq b_i, e_i \leq n$).

Гарантируется, что граф связан.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — сумму парных расстояний между вершинами.

Пример

sumdist.in	sumdist.out
5 5 1 2 2 3 3 4 5 3 1 5	16

Задача С. Кратчайший путь двух коней [0.3 сек, 256 mb]

Переведите каждого из двух коней из одной клетки в другую за наименьшее общее число ходов. Два коня не могут одновременно находиться в одной клетке.

Формат входных данных

Во входном файле записаны координаты первого и второго коня, затем координаты клеток, куда нужно их переместить.

Формат выходных данных

Программа должна вывести последовательность ходов коней в виде нескольких строк. Первым символом в строке должен быть номер коня (1 или 2), затем, через пробел, координаты клетки, в которую он переставляется. Необходимо вывести любое из возможных оптимальных решений.

Пример

knight2.in	knight2.out
a1	1 b3
c2	1 d4
c2	2 a1
a1	1 c2

Задача D. Island. Островные государства [0.5 сек, 256 mb]

Суровые феодальные времена переживала некогда великая островная страна Байтландия. За главенство над всем островом борются два самых сильных барона. Таким образом, каждый город страны контролируется одним из правителей. Как водится издревле, некоторые из городов соединены двусторонними дорогами. Бароны очень не любят друг друга и стараются делать как можно больше пакостей. В частности, теперь для того чтобы пройти по дороге, соединяющей города различных правителей, надо заплатить пошлину — один байтландский рубль.

Программист Вася живет в городе номер 1. С наступлением лета он собирается съездить в город N на Всебайтландское сборище программистов. Разумеется, он хочет затратить при этом как можно меньше денег и помочь ему здесь, как обычно, предлагается Вам.

Формат входных данных

В первой строке входного файла записано два числа N и M ($1 \leq N, M \leq 100\,000$) — количество городов и количество дорог соответственно.

В следующей строке содержится информация о городах — N чисел 1 или 2 — какому из баронов принадлежит соответствующий город.

В последних M строках записаны пары $1 \leq a, b \leq N, a \neq b$. Каждая пара означает наличие дороги из города a в город b . По дорогам Байтландии можно двигаться в любом направлении.

Формат выходных данных

Если искомого пути не существует, выведите единственное слово `impossible`. В противном случае в первой строке напишите минимальную стоимость и количество посещенных городов, а во вторую выведите эти города в порядке посещения. Если минимальных путей несколько, выведите любой.

Пример

island.in	island.out
7 8 1 1 1 1 2 2 1 1 2 2 5 2 3 5 4 4 3 4 7 1 6 6 7	0 5 1 2 3 4 7
5 5 1 2 1 1 2 1 2 2 3 3 5 1 4 4 5	1 3 1 4 5

Задача E. Диаметр графа [0.3 sec, 256 mb]

Дан связный взвешенный неориентированный граф.

Рассмотрим пару вершин, расстояние между которыми максимально среди всех пар вершин. Расстояние между ними называется *диаметром графа*. *Эксцентриситетом вершины v* называется максимальное расстояние от вершины v до других вершин графа. *Радиусом графа* называется наименьший из эксцентриситетов вершин. Найдите диаметр и радиус графа.

Формат входных данных

В первой строке входного файла единственное число: N ($1 \leq N \leq 100$) — количество вершин графа. 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. На главной диагонали матрицы всегда нули; веса рёбер не превышают 1000.

Формат выходных данных

В выходной файл выведите два числа — диаметр и радиус графа.

Пример

diameter.in	diameter.out
4	8
0 -1 1 2	5
-1 0 -1 5	
1 -1 0 4	
2 5 4 0	

Задача F. Расстояние между вершинами [1 sec, 256 mb]

Дан неориентированный взвешенный граф. Найти вес минимального пути между двумя вершинами.

Формат входных данных

Первая строка входного файла содержит натуральные числа N , M , вторая строка содержит натуральные числа S и F ($N \leq 5\,000$, $M \leq 100\,000$, $1 \leq S, F \leq N$, $S \neq F$) — количество вершин и ребер графа а также номера вершин, длину пути между которыми требуется найти.

Следующие M строк по три натуральных числа b_i , e_i и w_i — номера концов i -ого ребра и его вес соответственно ($1 \leq b_i, e_i \leq n$, $0 \leq w_i \leq 100\,000$).

Формат выходных данных

Первая строка должна содержать одно натуральное число — вес минимального пути между вершинами S и F . Во второй строке через пробел выведите вершины на кратчайшем пути из S в F в порядке обхода.

Если путь из S в F не существует, выведите -1 .

Пример

distance.in	distance.out
4 4	3
1 3	1 2 3
1 2 1	
3 4 5	
3 2 2	
4 1 4	

Задача G. Расстояние между вершинами [1 sec, 256 mb]

Коль Дейкстру писать без кучи,
То тайм-лимит ты получишь...
А в совсем крутой задаче
Юзай кучу Фибоначчи!

Спектакль преподавателей
ЛКШ.июль-2007

Дан взвешенный граф. Требуется найти вес минимального пути между двумя вершинами.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно. Вторая строка входного файла содержит натуральные числа s и t — номера вершин, длину пути между которыми требуется найти ($1 \leq s, t \leq n$, $s \neq t$).

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 ($1 \leq b_i, e_i \leq n$, $0 \leq w_i \leq 100$).

$n \leq 100\,000$, $m \leq 200\,000$.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — вес минимального пути между вершинами s и t .

Если путь из s в t не существует, выведите -1.

Пример

distance.in	distance.out
4 4	3
1 3	
1 2 1	
3 4 5	
3 2 2	
4 1 4	

Задача H. Path. Кратчайший путь [1 sec, 256 mb]

Дан взвешенный ориентированный граф и вершина s в нем. Требуется для каждой вершины u найти длину кратчайшего пути из s в u .

Формат входных данных

Первая строка входного файла содержит n , m и s — количество вершин, ребер и номер выделенной вершины соответственно ($2 \leq n \leq 2000$, $1 \leq m \leq 6000$).

Следующие m строк содержат описание ребер. Каждое ребро задается стартовой вершиной, конечной вершиной и весом ребра. Вес каждого ребра — целое число, не превосходящее 10^{15} по модулю. В графе могут быть кратные ребра и петли.

Формат выходных данных

Выведите n строк — для каждой вершины u выведите длину кратчайшего пути из s в u , '*' если не существует путь из s в u и '-' если не существует кратчайший путь из s в u .

Пример

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

Бонус

Задача I. Кратчайший путь [2 сек, 256 mb]

Надеюсь, все вы умеете искать в ориентированном графе кратчайший путь. В этой задаче вам предлагается свое умение продемонстрировать.

Вам дан ориентированный взвешенный граф. Веса ребер — целые числа от 1000 до 2000. Нужно несколько раз (не более 1000) ответить на следующий запрос: длина кратчайшего пути из некоторой вершины s в некоторую вершину t .

Формат входных данных

На первой строке числа N и M ($1 \leq N \leq 10^5$, $0 \leq M \leq 2 \cdot 10^5$) — количество вершин и ребер нашего графа, соответственно. Вершины нумеруются целыми числами от 1 до N . Далее M строк содержат информацию о ребрах графа. Каждое ребро задается тремя числами — номер начала, номер конца и вес. Все веса — целые числа от 1000 до 2000. В графе могут быть и петли, и кратные ребра. Следующая строка содержит число K ($1 \leq K \leq 10^3$) — количество запросов. В следующих K строках задаются запросы. Каждый запрос описывается двумя числами — из какой вершины, и в какую должен вести путь.

Формат выходных данных

Для каждого запроса выведите на отдельной строке целое число — длину кратчайшего пути. Если кратчайшего пути не существует следует вывести -1 .

Пример

shortest.in	shortest.out
5 5	-1
1 2 2000	3000
1 3 1000	0
1 4 1200	2000
2 3 1500	
3 4 1500	
4	
1 5	
2 4	
3 3	
1 2	

Замечание

Путем в графе называется такая последовательность ребер, что конец i -го совпадает с началом $i+1$ -го. Длиной пути называется суммарный вес ребер. Путь является кратчайшим, если его длина минимальна.

Задача J. Экскурсия [0.2 sec, 256 mb]

ЛКШата собираются на экскурсию в Кострому. Кострома — красивый старый город, в котором площади соединяются друг с другом короткими улицами, каждую из которых можно пройти не более чем за десять минут. ЛКШата хотят составить интересный маршрут экскурсии. Так как они поедут на автобусах, то маршрут экскурсии должен начинаться и заканчиваться на одной и той же площади. К сожалению, у ЛКШат будет очень мало времени. Поэтому они решили выбрать наиболее короткий кольцевой маршрут, не проходящий ни по какой улице дважды.

Помогите ЛКШатам найти такой маршрут.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество площадей и улиц в Костроме ($1 \leq n \leq 1000$, $1 \leq m \leq 10000$). Площади занумерованы от 1 до n .

Последующие m строк содержат описания улиц. Каждая улица описывается тремя целыми числами — номерами площадей, которые она соединяет, и количеством минут, которые требуются ЛКШатам на то, чтобы пройти по ней (от одной до десяти минут). Между двумя площадями может быть более одной улицы. Улица соединяет две различные площади.

Гарантируется, что в Костроме существует как минимум один кольцевой маршрут.

Формат выходных данных

Первая строка выходного файла должна содержать единственное число — продолжительность минимального маршрута в минутах.

Пример

excurs.in	excurs.out
5 6 1 2 1 2 3 10 1 3 1 2 4 1 3 4 1 1 5 1	4

Задача К. Грязь [0.15 сек, 256 mb]

— Здравствуйте! Могу я поговорить с Петровым? Алё, милый, привет... ты знаешь, у нас дома небольшая авария произошла... Но твой компьютер не пострадал, не волнуйся. Но теперь там немного грязно. Ну, то есть очень грязно. Но ты не волнуйся, я приготовила тебе твои болотные сапоги, у входа стоят. А грязь я уберу, как будет свободное время. Когда? Ну, наверное, когда в отпуск пойду. А, ну когда вернёмся из Турции. А, ну значит в следующий отпуск, но обязательно уберу. А пока я у мамы поживу. И ты, кстати, тоже можешь. Ну, как хочешь, я же не заставляю... Только пока я не убрала, ты там грязь не разводи, сильно сапогами по грязи не шлёпай и когда по чистому ходишь, сапоги снимай и тапочки обувай, я их тоже возле входа поставила, ты их бери с собой, когда идёшь по грязи и переобувай. А когда по чистому идёшь, бери сапоги, там грязь в разных местах. Программисты, как известно, не самые трудолюбивые люди, поэтому убирать грязь не станут. Но переобувать болотные сапоги каждый раз, когда переходишь от грязного пола к чистому и наоборот — удовольствие ниже среднего, уж лучше пройти лишние несколько метров. Чтобы прожить время до следующего отпуска с комфортом, надо срочно выработать способ добираться из одной точки квартиры с минимальным количеством переобуваний по пути, ну а уж среди них, конечно, выбрать самый короткий.

Формат входных данных

В первой строке даны два целых числа M и N — размеры квартиры (в у.е.). $1 \leq N, M \leq 500$. Два целых числа во второй строке — координаты компьютера (в у.е.), а два целых числа в третьей строке — координаты холодильника (тоже в у.е.). Далее идут M строк по N символов в каждой — план квартиры. На плане 1 означает чистое место, 2 — грязное, 0 — стена или непроходимая грязь. Переходить можно только на клетки, имеющие общую вершину с данной, при переходе с чистой на грязную и наоборот надо переобуваться. Холодильник и компьютер находятся не в клетках, помеченных нулём. Левая верхняя клетка плана имеет координаты (1, 1).

Формат выходных данных

Длина кратчайшего пути (количество преодолённых квадратиков, включая начальный и конечный) с минимальным количеством переобуваний, и, через пробел, количество переобуваний (переобувание проходит при переходе с грязного на чистое и наоборот). Если пройти к холодильнику невозможно, вывести числа 0 0.

Пример

stdin	stdout
3 7 1 1 3 7 1200121 1212020 1112021	8 4

Задача L. Таможенные пошлины [1 sec, 256 mb]

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 ($2 \leq N \leq 10\,000$, $1 \leq M \leq 100\,000$), 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 ($1 \leq X, Y \leq N$; $X \neq Y$; $1 \leq Z \leq 1\,000\,000\,000$), 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 ($1 \leq A, B \leq N$; $A \neq B$) — 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

customs.in	customs.out
5 6	6
1 2 10	
1 3 4	
3 2 3	
1 4 1	
4 5 2	
5 2 3	
1 2	