

Задача А. Лабиринт знаний

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

В ЛКШ построили аттракцион «Лабиринт знаний». Лабиринт представляет собой N комнат, занумерованных от 1 до N , между некоторыми из которых есть двери. Когда человек проходит через дверь, показатель его знаний изменяется на определенную величину, фиксированную для данной двери. Вход в лабиринт находится в комнате 1, выход — в комнате N . Каждый ЛКШонок проходит лабиринт ровно один раз и попадает в группу в зависимости от набранных знаний (при входе в лабиринт этот показатель равен нулю). Ваша задача показать наилучший результат.

Формат входного файла

Первая строка входного файла содержит целые числа N ($1 \leq N \leq 2000$) — количество комнат и M ($0 \leq M \leq 10000$) — количество дверей. В каждой из следующих M строк содержится описание двери — номера комнат, из которой она ведет и в которую она ведет, а также целое число, которое прибавляется к количеству знаний при прохождении через дверь (это число по модулю не превышает 10000). Двери могут вести из комнаты в нее саму, между двумя комнатами может быть более одной двери.

Формат выходного файла

В выходной файл выведите «:» — если можно получить неограниченно большой запас знаний, «:(» — если лабиринт пройти нельзя, и максимальное количество набранных знаний в оставшемся случае.

Примеры

<code>maze.in</code>	<code>maze.out</code>
2 2 1 2 3 1 2 7	7

Задача В. Рейсы во времени

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

Между N населёнными пунктами совершаются пассажирские рейсы на машинах времени.

В момент времени 0 вы находитесь в пункте A . Вам дано расписание рейсов. Требуется оказаться в пункте B как можно раньше (то есть в наименьший возможный момент времени).

При этом разрешается делать пересадки с одного рейса на другой. Если вы прибываете в некоторый пункт в момент времени T , то вы можете уехать из него любым рейсом, который отправляется из этого пункта в момент времени T или позднее (но не раньше).

Формат входного файла

Первая строка входного файла содержит число N — количество населённых пунктов ($1 \leq N \leq 1000$). Вторая строка содержит два числа A и B — номера начального и конечного пунктов. Третья строка содержит число K — количество рейсов ($0 \leq K \leq 1000$). Следующие K строк содержат описания рейсов, по одному на строке. Каждое описание представляет собой четвёрку целых чисел. Первое число каждой четвёрки задаёт номер пункта отправления, второе — время отправления, третье — пункт назначения, четвертое — время прибытия. Номера пунктов — натуральные числа из диапазона от 1 до N . Пункт назначения и пункт отправления могут совпадать. Время измеряется в некоторых абсолютных единицах и задаётся целым числом, по модулю не превышающим 10^9 . Поскольку рейсы совершаются на машинах времени, то время прибытия может быть как больше времени отправления, так и меньше, или равным ему.

Гарантируется, что входные данные таковы, что добраться из пункта A в пункт B всегда можно.

Формат выходного файла

Выведите в выходной файл минимальное время, когда вы сможете оказаться в пункте B .

Примеры

<code>time.in</code>	<code>time.out</code>
2 1 1 2 1 1 2 10 1 10 1 9	0
1 1 1 3 1 3 1 -5 1 -5 1 -3 1 -4 1 -10	-10

Задача С. Паросочетание

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

Двудольным графом называется неориентированный граф (V, E) , $E \subseteq V \times V$ такой, что его множество вершин V можно разбить на два множества A и B , для которых $\forall (e_1, e_2) \in E$

$e_1 \in A$, $e_2 \in B$ и $A \cup B = V$, $A \cap B = \emptyset$.

Паросочетанием в двудольном графе называется любой набор его несмежных рёбер, то есть такой набор $S \subseteq E$, что для любых двух рёбер $e_1 = (u_1, v_1)$, $e_2 = (u_2, v_2)$ из S $u_1 \neq u_2$ и $v_1 \neq v_2$.

Ваша задача — найти максимальное паросочетание в двудольном графе, то есть паросочетание с максимально возможным числом рёбер.

Формат входного файла

В первой строке записаны два целых числа n и m ($1 \leq n, m \leq 250$), где n — число вершин в множестве A , а m — число вершин в B .

Далее следуют n строк с описаниями рёбер — i -я вершина из A описана в $(i+1)$ -й строке файла. Каждая из этих строк содержит номера вершин из B , соединённых с i -й вершиной A . Гарантируется, что в графе нет кратных ребер. Вершины в A и B нумеруются независимо (с единицы). Список завершается числом 0.

Формат выходного файла

Первая строка выходного файла должна содержать одно целое число l — количество рёбер в максимальном паросочетании. Далее следуют l строк, в каждой из которых должны быть два целых числа u_j и v_j — концы рёбер паросочетания в A и B соответственно.

Пример

pairs.in	pairs.out
2 2	2
1 2 0	1 1
2 0	2 2

Задача D. Замощение доминошками

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

Дано игровое поле размера $n \times m$, некоторые клетки которого уже замощены. Замостить свободные соседние клетки поля доминошкой размера 1×2 стоит a условных единиц, а замостить свободную клетку поля квадратиком размера 1×1 — b условных единиц.

Определите, какая минимальная сумма денег нужна, чтобы замостить всё поле.

Формат входного файла

Первая строка входного файла содержит 4 целых числа n , m , a , b ($1 \leq n, m \leq 100$, $|a| \leq 1000$, $|b| \leq 1000$). Каждая из последующих n строк содержит по m символов: символ "." (точка) обозначает занятую клетку поля, а символ "*" (звёздочка) — свободную.

Формат выходного файла

В выходной файл выведите одно число — минимальную сумму денег, имея которую можно замостить свободные клетки поля (и только их).

Пример

dominoes.in	dominoes.out
2 3 3 2 .** .*	5

Задача E. Покрытие путями

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

Задан ориентированный ациклический граф. Требуется определить минимальное количество не пересекающихся по вершинам путей, покрывающих все вершины.

Формат входного файла

Первая строка входного файла содержит целые числа n и m — количества вершин и рёбер графа соответственно ($2 \leq n \leq 1000$, $0 \leq m \leq 10^5$). В следующих m строках содержатся по два натуральных числа — номера вершин u и v , которые соединяет ребро (u, v) .

Формат выходного файла

В первой строке выходного файла выведите натуральное число k — минимальное количество путей, необходимых для покрытия всех вершин.

Пример

paths.in	paths.out
3 3 1 3 3 2 1 2	1