

Задача А. Метро

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

Метрополитен состоит из нескольких линий метро. Все станции метро в городе пронумерованы натуральными числами от 1 до N . На каждой линии расположено несколько станций. Если одна и та же станция расположена сразу на нескольких линиях, то она является станцией пересадки и на этой станции можно пересесть с любой линии, которая через нее проходит, на любую другую (опять же проходящую через нее).

Напишите программу, которая по данному вам описанию метрополитена определит, с каким минимальным числом пересадок можно добраться со станции А на станцию В. Если данный метрополитен не соединяет все линии в одну систему, то может так получиться, что со станции А на станцию В добраться невозможно, в этом случае ваша программа должна это определить.

Формат входного файла

Сначала вводится число N — количество станций метро в городе ($2 \leq N \leq 100$). Далее следует число M — количество линий метро ($1 \leq M \leq 20$). Далее идет описание M линий. Описание каждой линии состоит из числа P_i — количество станций на этой линии ($2 \leq P_i \leq 50$) и P_i чисел, задающих номера станций, через которые проходит линия (ни через какую станцию линия не проходит дважды).

Затем вводятся два различных числа: А — номер начальной станции, и В — номер станции, на которую нам нужно попасть. При этом если через станцию А проходит несколько линий, то мы можем спуститься на любую из них. Так же если через станцию В проходит несколько линий, то нам не важно, по какой линии мы приедем.

Формат выходного файла

Выведите минимальное количество пересадок, которое нам понадобится. Если добраться со станции А на станцию В невозможно, программа должна вывести одно число -1 (минус один).

Пример

| metro.in | metro.out |
|--|-----------|
| 5 2 4 1 2 3 4 2 5 3 3 1 | 0 |
| 5 5 2 1 2 2 1 3 2 2 3 2 3 4 2 4 5 1 5 | 2 |
| 10 2 6 1 3 5 7 4 9 6 2 4 6 8 10 7 3 8 | 1 |
| 4 2 2 1 2 2 3 4 1 3 | -1 |

Задача В. Домой на электричках

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

Одна из команд-участниц олимпиады решила вернуться домой на электричках. При этом ребята хотят попасть домой как можно раньше. К сожалению, не все электрички идут от города, где проводится олимпиада, до станции, на которой живут ребята. И, что еще более обидно, не все электрички, которые идут мимо их станции, останавливаются на ней (равно как вообще, электрички останавливаются далеко не на всех станциях, мимо которых они идут).

Все станции на линии пронумерованы числами от 1 до N . При этом станция номер 1 находится в городе, где проводится олимпиада, и в момент времени 0 ребята приходят на станцию. Станция, на которую нужно попасть ребятам, имеет номер E .

Напишите программу, которая по данному расписанию движения электричек вычисляет минимальное время, когда ребята могут оказаться дома.

Формат входного файла

Во входном файле записаны сначала числа N ($2 \leq N \leq 100$) и E ($2 \leq E \leq N$). Затем записано число M ($0 \leq M \leq 100$), обозначающее число рейсов электричек. Далее идет описание M рейсов электричек. Описание каждого рейса электрички начинается с числа K_i ($2 \leq K_i \leq N$) — количества станций, на которых она останавливается, а далее следует K_i пар чисел, первое число каждой пары задает номер станции, второе — время, когда электричка останавливается на этой станции (время выражается целым числом из диапазона от 0 до 10^9). Станции внутри одного рейса упорядочены в порядке возрастания времени. В течение одного рейса электричка все время движется в одном направлении — либо от города, либо к городу.

Формат выходного файла

В выходной файл выведите одно число — минимальное время, когда ребята смогут оказаться на своей станции. Если существующими рейсами электричек они добраться не смогут, выведите -1 .

Пример

| rail.in | rail.out |
|----------------------|----------|
| 5 3 | 20 |
| 4 | |
| 2 1 5 2 10 | |
| 2 2 10 4 15 | |
| 4 5 0 4 17 3 20 2 35 | |
| 3 1 2 3 40 4 45 | |

Задача С. Два профессора

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

Профессор Флойд и профессор Дейкстра ненавидят друг друга. После переезда университета во вновь отстроенный университетский городок они потребовали себе

кабинеты в зданиях, максимально удалённых друг от друга. Вам поручено найти найти расстояние между двумя такими зданиями.

Иными словами, требуется найти два здания, кратчайший путь между которыми наибольший среди всех пар зданий, и вывести длину этого пути. Так как профессорам иногда все же нужно встречаться, путь между выбранными зданиями должен существовать.

Формат входного файла

В первой строке находятся два числа N и M — количество зданий и количество дорог, соединяющих здания ($1 \leq N \leq 1000$; $M \leq N * (N - 1)/2$). Далее в M строках расположены описания дорог: 3 целых числа s_i, e_i, l_i — здания, в которых начинается и заканчивается дорога и длина дороги соответственно ($1 \leq s_i, e_i \leq N$; $0 \leq l_i \leq 100$), дороги двунаправленные).

Формат выходного файла

Необходимо вывести одно число — искомое расстояние.

Пример

| professors.in | professors.out |
|---------------|----------------|
| 3 2 | 3 |
| 1 2 1 | |
| 2 3 2 | |
| 3 0 | 0 |

Задача D. Производство деталей

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

Предприятие «Авто-2010» выпускает двигатели для известных во всем мире автомобилей. Двигатель состоит ровно из n деталей, пронумерованных от 1 до n , при этом деталь с номером i изготавливается за p_i секунд. Специфика предприятия «Авто-2010» заключается в том, что там одновременно может изготавливаться лишь одна деталь двигателя. Для производства некоторых деталей необходимо иметь предварительно изготовленный набор других деталей. Генеральный директор «Авто-2010» поставил перед предприятием амбициозную задачу — за наименьшее время изготовить деталь с номером 1, чтобы представить ее на выставке. Требуется написать программу, которая по заданным

зависимостям порядка производства между деталями найдет наименьшее время, за которое можно произвести деталь с номером 1.

Формат входного файла

Первая строка входного файла содержит число n ($1 \leq n \leq 100000$) — количество деталей двигателя. Вторая строка содержит n натуральных чисел p_1, p_2, \dots, p_n , определяющих время изготовления каждой детали в секундах. Время для изготовления каждой детали не превосходит 10^9 секунд. Каждая из последующих n строк входного файла описывает характеристики производства деталей. Здесь i -ая строка содержит число деталей k_i , которые требуются для производства детали с номером i , а также их номера. Сумма всех чисел k_i не превосходит 200000. Известно, что не существует циклических зависимостей в производстве деталей.

Формат выходного файла

В первой строке выходного файла должны содержаться два числа: минимальное время (в секундах), необходимое для скорейшего производства детали с номером 1 и число k деталей, которые необходимо для этого произвести. Во второй строке требуется вывести через пробел k чисел — номера деталей в том порядке, в котором следует их производить для скорейшего производства детали с номером 1.

Пример

| details.in | details.out |
|---|-----------------------|
| 3 100 200 300 1 2 0 2 2 1 | 300 2 2 1 |
| 2 2 3 1 2 0 | 5 2 2 1 |
| 4 2 3 4 5 2 3 2 1 3 0 2 1 3 | 9 3 3 2 1 |
| 3 1000000000 1000000000 1000000000 2 2 3 0 0 | 3000000000 3 3 2 1 |

Задача Е. Компоненты связности

Имя входного файла: matrix.in
Имя выходного файла: matrix.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Дан неориентированный невзвешенный граф. Необходимо посчитать количество его компонент связности.

Формат входного файла

В первой строке входного файла содержится одно натуральное число N ($N \leq 100$) — количество вершин в графе. Далее в N строках по N чисел — матрица смежности графа: в i -ой строке на j -ом месте стоит 1, если вершины i и j соединены ребром, и 0, если ребра между ними нет. На главной диагонали матрицы стоят нули. Матрица симметрична относительно главной диагонали.

Формат выходного файла

Вывести одно целое число — искомое количество компонент связности графа.

Пример

| matrix.in | matrix.out |
|---|------------|
| 6 0 1 1 0 0 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 | 3 |

Задача F. Коровы в стойла!

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

На прямой расположены стойла, в которые необходимо расставить коров так, чтобы минимальное расстояние между коровами было как можно больше.

Формат входного файла

В первой строке вводятся числа $N(2 \leq N \leq 10001)$ — количество стойл и $K(1 \leq K \leq N)$ — количество коров. Во второй строке задаются N натуральных чисел в порядке возрастания — координаты стойл (координаты не превосходят 10^9).

Формат выходного файла

Выведите одно число — наибольшее возможное допустимое расстояние.

Пример

| cows.in | cows.out |
|-----------------------|----------|
| 5 3 1 2 3 100 1000 | 99 |

Задача G. Табличка

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

Дана таблица, состоящая из N строк и M столбцов. В каждой клетке таблицы записано одно из чисел: 0 или 1. Расстоянием между клетками $(x1, y1)$ и $(x2, y2)$ назовем сумму $|x1 - x2| + |y1 - y2|$. Вам необходимо построить таблицу, в клетке (i, j) которой будет записано минимальное расстояние между клеткой (i, j) начальной таблицы и клеткой, в которой записана 1. Гарантируется, что хотя бы одна 1 в таблице есть.

Формат входного файла

В первой строке входного файла содержатся два натуральных числа N и M , не превосходящих 100. Далее идут N строк по M чисел - элементы таблицы.

Формат выходного файла

Выходной файл должен содержать N строк по M чисел — элементы искомой таблицы.

Пример

| table.in | table.out |
|-----------------------|----------------|
| 2 3 0 0 1 1 0 0 | 1 1 0 0 1 1 |