

Задача А. 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 входных данных

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 выходных данных

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

<code>pairs.in</code>	<code>pairs.out</code>
2 2	2
1 2 0	1 1
2 0	2 2

Задача В. Минимальное контролирующее множество

Имя входного файла: `min-dominating-set.in`
Имя выходного файла: `min-dominating-set.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Множество вершин V' графа $G = (V, E)$, содержащееся в V , называется контролирующим, если каждое ребро графа инцидентно хотя бы одной вершине из V' . Тривиальным примером контролирующего множества является множество всех вершин V . Ваша задача состоит в том, чтобы в данном двудольном графе найти какое-либо контролирующее множество вершин минимальной мощности.

Формат входных данных

В первой строке входного файла содержатся два числа n и m — количество вершин в первой и второй долях ($1 \leq n, m \leq 200$). Далее до конца файла в каждой строке содержатся номера смежных вершин из первой и второй доли. В графе не бывает кратных рёбер.

Формат выходных данных

В выходной файл в первой строке выведите мощность минимального контролирующего множества. Во второй строке выведите через пробел количества вершин из первой и второй долей, входящих в контролирующее множество. В третьей строке через пробел выведите номера вершин первой доли, входящих в контролирующее множество, а в четвертой — номера вершин второй доли, входящих в это множество.

Примеры

<code>min-dominating-set.in</code>	<code>min-dominating-set.out</code>
6 6 1 1 2 5 3 4 4 4 5 2 5 3 6 1 6 6	5 4 1 1 2 5 6 4
6 6 1 1 1 2 2 2 3 2 3 3 4 3 4 4 4 5 4 6 5 2 6 3	4 2 2 1 4 2 3

Задача С. Такси

Имя входного файла: `taxi.in`
Имя выходного файла: `taxi.out`
Ограничение по времени: 0.5 секунда
Ограничение по памяти: 256 мегабайт

Управлять службой такси — совсем не простое дело. Помимо естественной необходимости централизованного управления машинами для того, чтобы обслуживать заказы по мере их поступления и как можно быстрее, нужно также планировать поездки для обслуживания тех клиентов, которые сделали заказы заранее.

В вашем распоряжении находится список заказов такси на следующий день. Вам необходимо минимизировать число машин такси, необходимых чтобы выполнить все заказы.

Для простоты будем считать, что план города представляет собой квадратную решетку. Адрес в городе будем обозначать парой целых чисел: x -координатой и y -координатой. Время, необходимое для того, чтобы добраться из точки с адресом (a, b) в точку (c, d) , равно $|a - c| + |b - d|$ минут. Машина такси может выполнить очередной заказ, либо если это первый ее заказ за день, либо она успевает приехать в начальную точку из предыдущей конечной хотя бы за минуту до указанного срока. Обратите внимание, что выполнение некоторых заказов может окончиться после полуночи.

Формат входных данных

В первой строке входного файла записано число заказов M ($0 < M < 500$). Последующие M строк описывают сами заказы, по одному в строке. Про каждый заказ указано время отправления в формате `hh:mm` (в интервале с `00:00` по `23:59`), координаты (a, b) точки отправления и координаты (c, d) точки назначения. Все координаты во входном файле неотрицательные и не превосходят 200. Заказы записаны упорядоченными по времени отправления.

Формат выходных данных

В выходной файл выведите единственное целое число — минимальное количество машин такси, необходимых для обслуживания всех заказов.

Примеры

<code>taxi.in</code>	<code>taxi.out</code>
2 08:00 10 11 9 16 08:07 9 16 10 11	1
2 08:00 10 11 9 16 08:06 9 16 10 11	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 входных данных

Первая строка входного файла содержит 4 целых числа n, m, a, b ($1 \leq n, m \leq 100, |a| \leq 1\,000, |b| \leq 1\,000$). Каждая из последующих n строк содержит по m символов: символ "." (точка) обозначает занятую клетку поля, а символ "*" (звёздочка) — свободную.

Формат выходных данных

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 Е. День рождения

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

Митя знаком с m юношами и n девушками и хочет пригласить часть из них на свой день рождения. Ему известно, с какими девушками знаком каждый юноша, и с какими юношами знакома каждая девушка. Он хочет добиться того, чтобы каждый приглашённый был знаком со всеми приглашёнными противоположного пола, пригласив при этом максимально возможное число своих знакомых. Помогите ему это сделать!

Формат входных данных

Входной файл состоит из одного или нескольких наборов входных данных. В первой строке входного файла записано число наборов k ($1 \leq k \leq 20$). В последующих строках записаны сами наборы входных данных.

В первой строке каждого набора задаются числа $0 \leq m \leq 150$ и $0 \leq n \leq 150$. Далее следуют m строк, в каждой из которых записано одно или несколько чисел — номера девушек, с которыми знаком i -й юноша (каждый номер встречается не более одного раза). Строка завершается числом 0.

Формат выходных данных

Для каждого набора выведите четыре строки. В первой из них выведите максимальное число знакомых, которых сможет пригласить Митя. В следующей строке выведите количество юношей и количество девушек в максимальном наборе знакомых. Следующие две строки должны содержать номера приглашённых юношей и приглашённых девушек соответственно. Если максимальных наборов несколько, то выведите любой из них.

Примеры

birthday.in	birthday.out
2	4
2 2	2 2
1 2 0	1 2
1 2 0	1 2
3 2	4
1 2 0	2 2
2 0	1 3
1 2 0	1 2

Задача F. Толстые хоббиты

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

Ни один хоббит не в состоянии в одиночку противостоять полчищам Мордора... В последний поход против Мордора Гэндальф решил отправить N хоббитов из Шира. Но часть хоббитов наотрез отказалась, жалуюсь на то, что другие хоббиты наверняка будут дразнить их толстыми. После опроса всех хоббитов оказалось, что любой хоббит отказывается принять участие в походе в том случае, если с ним в поход выступит хотя бы один хоббит с меньшим весом. К счастью для Средиземья, не все хоббиты знают свой точный вес. В Шире были всего одни весы чашечного типа, позволяющие для пары хоббитов определить, какой хоббит тяжелее. Некоторые пары хоббитов взвешивались на этих весах. Всем хоббитам известен результат всех взвешиваний. Гэндальф абсолютно уверен, что в Шире нет двух хоббитов одного веса. Он заинтересован в том, чтобы отряд состоял из наибольшего количества хоббитов. Однако найти наибольшее множество хоббитов, среди которых ни один не считает себя тяжелее другого, оказалось не так-то просто. Подскажите Гэндальфу, на сколько хоббитов он может рассчитывать. Помните при этом, что хоббиты умные существа и знают, что если Сэм тяжелее Пиппина, а Пиппин тяжелее Фродо, то Сэм и подавно будет тяжелее Фродо.

Формат входных данных

В первой строке дано целое число N – количество хоббитов ($2 \leq N \leq 100$). Все хоббиты пронумерованы целыми числами от 1 до N . В следующих N строках записана матрица размера $N \times N$. Если i -й и j -й хоббит взвешивались на чашечных весах и оказалось, что i -й хоббит тяжелее, то в i -й строке матрицы на j -й позиции стоит единица. Во всех остальных случаях в матрице стоят нули.

Формат выходных данных

В первой строке выведите размер наибольшего множества хоббитов, готового выступить в поход, во второй строке перечислите номера хоббитов из этого множества через пробел.

Примеры

<code>hobbits.in</code>	<code>hobbits.out</code>
2 0 1 0 0	1 2
3 0 0 0 0 0 0 0 0 0	3 1 2 3