Задача A. Максимальный поток

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф без петель и кратных ребер из N вершин (вершины нумеруются от 1 до N). Для каждого ребра известна его пропускная способность. Найти величину максимального потока из вершины 1 в вершину N. По каждому ребру поток может течь в любую сторону.

Формат входного файла

Два числа N и K ($2 \le N \le 100, 0 \le K \le \frac{n(n-1)}{2}$) - количество вершин и ребер в графе. K строк по три числа в каждой - $a, b, c (1 \le a, b \le N, 1 \le c \le 10^9)$ - номера вершин, соединенных ребром, и пропускная способность ребра.

Формат выходного файла

Одно число - величина максимального потока из вершины 1 в вершину N.

Пример

stdin	stdout
5 7 1 2 2 2 5 5 1 3 6 3 4 2 4 5 1 3 2 3 2 4 1	6

Задача B. Поток в двудольном графе

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан двудольный граф, исток и сток. Каждая доля состоит из N вершин. Из истока в левую долю ведут ребра пропускной способности a_i, из каждой вершины правой доли в сток ведут ребра пропускной способности b_i. Также есть ребра между вершинами левой и правой долей бесконечной пропускной способности, по которым поток может течь как в одну так и в другую сторону. Найти величину максимального потока из истока в сток.

Формат входного файла

В первой строке даны числа N и K ($1 \le N \le 10^4$, $0 \le K \le 10^5$) - количество вершин в каждой из долей и количество ребер между долями. Во второй строке перечислены числа a_i ($1 \le a_i \le 10^4$) - пропускные способности ребер из истока в каждую вершину левой доли. В третьей строке перечислены числа b_i ($1 \le b_i \le 10^4$) - пропускные способности ребер из каждой вершины правой доли в сток. В каждой из последующих K строк записаны по два числа u, v ($1 \le u, v \le N$) означающие наличие ребра и вершины u левой доли в вершину v правой доли.

Формат выходного файла

Вывести одно число - величину максимального потока.

Пример

stdin	stdout
3 4 3 2 1 5 4 4 1 1 1 2 2 3 3 3	6

Задача C. K-связность

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф без петель и кратных ребер. Найти минимальное количество ребер, которые нужно удалить, чтобы граф стал несвязным.

Формат входного файла

Два числа N и K ($2 \le N \le 100, 0 \le K \le \frac{n(n-1)}{2}$) - количество вершин и ребер в графе. K строк по два числа в каждой - a, b ($1 \le a, b \le N$) - номера вершин, соединенных ребром

Формат выходного файла

Одно число - минимальное число ребер, которые необходимо удалить, чтобы граф стал несвязным.

Пример

stdin	stdout
3 3 1 2 2 3 3 1	2

Задача D. Вершинно-непересекающиеся пути

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан ациклический ориенитрованный граф. Разбить граф на минимальное количество вершинно-непересекающихся путей. То есть каждая вершина должна принадлежать ровно одному пути.

Формат входного файла

Два числа N и K ($1 \le N \le 500$, $0 \le K \le \frac{n(n-1)}{2}$) - количество вершин и ребер в графе. К строк по два числа в каждой - a, b ($1 \le a, b \le N$), означающих наличие ребра из вершины a в вершину b. Каждая пара (a, b) встречается не более одного раза.

Формат выходного файла

Одно число - минимальное количество путей.

Пример

stdin	stdout
7 6 1 4 2 4 3 4 4 5 4 6 4 7	5

Задача E. ABC

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дано прямоугольное поле, состоящее из свободных клеток и занятых. Из каждой свободной клетки можно перейти в любую свободную, соседнюю с ней по стороне. Определить, можно ли пройти из клетки A в клетку C, пройдя через клетку B. Причем не разрешается выходить за пределы поля, проходить через занятые клетки и проходить более одного раза через одну клетку.

Формат входного файла

Два числа R и C ($1 \le R, C \le 100$) - размеры поля. Далее R строк по C символов в каждой - описание клеток. Каждая клетка может быть либо '.' (свободной), либо '#' (занятой), либо контрольной точкой 'A', 'B' или 'C'. Каждый из символов 'A', 'B' и 'C' встречается в описании ровно один раз.

Формат выходного файла

Вывести "Yes", если искомый путь существует, иначе - "No"

Пример

stdin	stdout
4 4 A... .##. .##B C...	Yes
4 4 A... C... .... ..#B	No

Задача F. Домино

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дано прямоугольное поле, состоящее из свободных клеток и занятых. Определить, можно ли замостить поле доминошками 2×1 таким образом, чтобы каждая пустая клетка была покрыта ровно одной доминошкой, каждая занятая была не покрыта, и никакая доминошка не выходила за пределы поля.

Формат входного файла

Два числа R и C ($1 \le R, C \le 100$) - размеры поля. Далее R строк по C символов в каждой - описание клеток. Каждая клетка может быть либо '.' (свободной), либо '#' (занятой).

Формат выходного файла

Вывести "Yes", если замощение возможно, иначе - "No"

Пример

stdin	stdout
3 3 ... .#. ...	Yes

Задача G. Черное и белое

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф без петель и кратных ребер. Вершины графа бывают двух типов - черные и белые. Если черная вершина не имеет ни одной соседней белой, то она сразу же исчезает. Аналогично, если белая вершина не имеет черных соседей, то она тоже пропадает. Для каждой вершины известна стоимость ее удаления. Определить минимальную стоимость удаления всех вершин.

Формат входного файла

Два числа N и K ($1 \le N \le 100, 0 \le K \le \frac{n(n-1)}{2}$) - количество вершин и ребер в графе. N строк по два числа в каждой t, c ($1 \le c \le 10^6$) - тип вершины (0 - белая, 1 - черная) и стоимость ее удаления. Далее K строк по два числа в каждой - a, b ($1 \le a, b \le N$) - номера вершин, соединенных ребром.

Формат выходного файла

Одно число - минимальная стоимость удаления всех вершин.

Пример

stdin	stdout
4 3 0 4 0 3 1 3 1 5 1 3 2 3 2 4	6

Задача H. Билет на катер

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Есть несколько катеров и большая группа людей. Для каждого катера известны его вместимость и цена на билет. Для каждого человека известно, на каких катерах он предпочел бы прокатиться. Требуется распределить людей по катерам на одну морскую прогулку так, чтобы суммарная стоимость, потраченная на билеты, была минимальна.

Формат входного файла

Два числа N и K ($1 \le N \le 16, 1 \le K \le 100$) - количество катеров и людей. N строк по два числа в каждой - s, c ($0 \le s \le 100, 0 \le c \le 10^6$) - количество мест в катере и стоимость одного билета. Далее K строк, в каждой из которых дано число t - количество катеров, которые подходят текущему человеку, после чего следует список номеров этих катеров. Все номера в пределах от 1 до N, каждый номер встречается не более одного раза.

Формат выходного файла

Одно число - суммарная стоимость одной прогулки, либо -1, если всех людей рассадить по катерам не получится.

Пример

stdin	stdout
3 5 1 1 2 2 3 3 3 1 2 3 2 1 2 1 1 2 1 2 3 1 2 3	11

Задача I. Прогулка на катере

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Есть несколько катеров и большая группа людей. Для каждого катера известна его вместимость. Для каждого человека известно, на каких катерах он предпочел бы прокатиться. После опыта предыдущей задачи владельцы катеров выяснили, что подобный способ оплаты может быть весьма невыгоден. Поэтому было решено брать одинаковую плату за каждый используемый катер, независимо от количества людей на нем. Требуется распределить людей по катерам на одну морскую прогулку так, чтобы минимизировать количество используемых катеров.

Формат входного файла

Два числа N и K ($1 \le N \le 16, 1 \le K \le 100$) - количество катеров и людей. N строк по одному числу в каждой - s ($0 \le s \le 100$) - количество мест в катере. Далее K строк, в каждой из которых дано число t - количество катеров, которые подходят текущему человеку, после чего следует список номеров этих катеров. Все номера в пределах от 1 до N, каждый номер встречается не более одного раза.

Формат выходного файла

Одно число - минимальное количество катеров, либо -1, если всех людей рассадить по катерам не получится.

Пример

stdin	stdout
3 5 1 2 3 3 1 2 3 2 1 2 1 1 2 1 2 3 1 2 3	3

Задача J. Доверие

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Большая группа людей решила порешать задачи. Каждый человек получил некоторое количество задач. При этом между некоторыми людьми существует особый вид взаимоотношений - доверие. Один из них может доверить некоторые свои задачи другому, но не наоборот. Отношения настолько доверительные, что одну задачу можно передоверять сколько угодно раз. Коллектив настолько сплоченный, что требуется определить минимальное возможное количество задач, которое будет у самого загруженного человека (того, у которого больше всего задач) после оптимального перераспределения задач.

Формат входного файла

Два числа N и K ($1 \le N \le 100, 0 \le K \le \frac{N \cdot (N-1)}{2}$) - количество людей и отношений. N чисел в одной строке ($1 \le v_i \le 10^6$) - исходное количество задач у каждого человека. Далее K строк по два числа в каждой - a, b ($1 \le a, b \le N$), означающие, что человек а может доверять задачи человеку b.

Формат выходного файла

Одно число - количество задач у самого загруженного человека.

Пример

stdin	stdout
4 3 10 6 3 1 1 2 2 3 3 4	5

Задача K. Змейка

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Есть прямоугольное игровое поле, разбитое на клетки. Некоторые клетки свободные, некоторые занятые. Допустим, в одну из свободных клеток поместили змейку размера 1. Первый игрок делает свой ход: он может увеличить змейку, присоединив к ней одну из соседних по стороне свободных клеток. Далее ходит второй игрок и т. д. Каждый игрок может присоединить любую свободную клетку, которая является соседней к последней присоединенной клетке. Уже присоединенные клетки считаются занятыми. Проигрывает тот, кто не может ходить. Для каждой свободной клетки поля определить, выиграет ли первый игрок, если изначально поместить змейку в эту клетку.

Формат входного файла

Два числа R и C ($1 \le R, C \le 100$) - размеры поля. Далее R строк по C символов каждая. Символы могут быть либо '.' - свободная клетка, либо '#' - занятая.

Формат выходного файла

Вывести R строк по C символов - игровое поле. Все символы '.' заменить на 'W', если клетка выигрышная для первого игрока, или на 'L', если проигрышная.

Пример

stdin	stdout
3 3	LWL WLW LWL

Задача L. Камень-ножницы-бумага

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

В знаменитой игре камень-ножницы-бумага довольно простые правила. Каждый из двух игроков загадывает одно из трех положений руки: камень, ножницы или бумагу. После чего оба игрока одновременно показывают свое положение, и дальше действуют такие правила определения победителя: если положения совпали (например, камень-камень), то ничья, иначе камень побеждает ножницы, ножницы побеждают бумагу, а бумага - камень.

В некоторой модификации игры участвует большое количество предметов, и уже непонятно, действительно ли такая игра честная. То есть нет ли такого, что одни положения более выгодны чем другие. Будем считать, что игра честная, если каждое положение имеет ничью с самим собой, выигрывает ровно у половины остальных и проигрывает другой половине. Дано описание игры, определить минимальное количество пар положений, для которых нужно изменить направление выигрыша, чтобы игра стала честной.

Формат входного файла

Нечетное число N ($1 \le N \le 100$) - количество предметов. Далее матрица $N \times N$, описывающая исходы для разных пар предметов. 1 соответствует выигрышу, 0 - ничье, -1 - проигрышу. На главной диагонали всегда стоят 0. Если в некоторой ячейке (i, j) стоит 1, то в (j, i) обязательно стоит -1.

Формат выходного файла

Минимальное количество пар предметов, для которых надо поменять победителя, чтобы игра стала честной.

Пример

stdin	stdout
3 0 1 1 -1 0 1 -1 -1 0	1

Задача M. Монотонность

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Пусть есть некотороая булевая функция, определенная на некотором множестве булевых векторов размерности K. Один вектор является предшествующим по отношению к другому, если они различны и каждая координата первого вектора не больше чем соответствующая координата второго. Функция называется монотонной, если ее значение на каждом векторе не меньше чем на любом предшествующем векторе.

Требуется переопределить функцию в минимальном количество векторов, чтобы она стала монотонной.

Формат входного файла

Два числа N и k ($1 \le N \le 1000, 1 \le K \le 10$) - количество векторов и размерность. Далее N строк по $k+1$ числу в каждой - булевый вектор и значение функции в нем. Все числа - 0 или 1. Все вектора различны.

Формат выходного файла

В первой строке минимальное количество векторов, значение функции в которых необходимо изменить, чтобы она стала монотонной. Во второй строке номера этих векторов без повторений в любом порядке.

Пример

stdin	stdout
8 3 0 0 0 1 0 0 1 0 0 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 1 1 0 1 1 1 1 0	2 1 8

Задача N. Обед

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Большая группа людей решила сходить на обед. Поскольку люди в группе решают задачи с разной скоростью, то в столовую они пришли в разное время. Для каждого человека известно, ко скольки он пришел в столовую и сколько блюд заказал. В столовой работают несколько официантов, каждый из них способен подать одно блюдо в минуту любому посетителю. Требуется рассчитать работу официантов таким образом, чтобы максимальное время ожидания еды было как можно меньше. Под ожиданием еды понимается разница во времени между моментом, когда человек пришел в столовую и когда ему подали все его блюда. Даже после окончания дня официанты могут продолжать разносить еду.

Формат входного файла

В первой строке два числа N, K ($1 \le N \le 100, 1 \le K \le 10$) - количество людей в группе и количество официантов. Далее N строк по два целых числа в каждой t и d ($0 \le t < 24*60, 1 \le d \le 10$) - время в минутах, в которое пришел человек, и сколько блюд он заказал.

Формат выходного файла

Одно число - минимум максимального времени ожидания.

Пример

stdin	stdout
4 2 0 7 2 2 3 3 5 1	2 4