

Задача А. Максимальный поток

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

Задан ориентированный граф, каждое ребро которого обладает целочисленной пропускной способностью. Найдите максимальный поток из вершины с номером 1 в вершину с номером n .

Формат входных данных

Первая строка входного файла содержит n и m — количество вершин и количество ребер графа ($2 \leq n \leq 100$, $1 \leq m \leq 1000$). Следующие m строк содержат по три числа: номера вершин, которые соединяет соответствующее ребро графа и его пропускную способность. Пропускные способности не превосходят 10^5 .

Формат выходных данных

В выходной файл выведите одно число — величину максимального потока из вершины с номером 1 в вершину с номером n .

Примеры

<code>maxflow.in</code>	<code>maxflow.out</code>
4 5 1 2 1 1 3 2 3 2 1 2 4 2 3 4 1	3

Задача В. Максимальный поток

Имя входного файла: flow2.in
Имя выходного файла: flow2.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Вам задан ориентированный граф G . Каждое ребро имеет некоторую пропускную способность. Найдите максимальный поток между вершинами 1 и n .

Формат входных данных

Первая строка входного файла содержит n и m — число вершин и ребер в графе ($2 \leq n \leq 500$, $1 \leq m \leq 10\,000$). Последующие строки описывают ребра. Каждое ребро задается тремя числами: начальная вершина ребра, конечная вершина ребра и пропускная способность ребра. Пропускные способности не превосходят 10^9 .

Формат выходных данных

Выведите величину максимального потока между вершинами 1 и n .

Далее для каждого ребра выведите величину потока, текущую по этому ребру.

Примеры

flow2.in	flow2.out
4 5 1 2 1 1 3 2 3 2 1 2 4 2 3 4 1	3 1 2 1 2 1
4 5 1 2 2 1 3 2 2 3 1 2 4 2 3 4 2	4 2 2 0 2 2
4 3 1 2 1 1 3 1 1 4 4	4 0 0 4

Задача С. Разрезание графа

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

Разбейте множество вершин заданного графа на два непустых подмножества A и B так, чтобы количество рёбер между вершинами различных подмножеств было минимально.

Формат входных данных

В первой строке входного файла записано целое число n ($2 \leq n \leq 50$) — число вершин в графе. Каждая из следующих n строк содержит по n символов. i -ый символ j -ой из этих строк равен “1”, если между вершинами i и j есть ребро, и “0” в противном случае. Заданная таким образом матрица смежности графа является антирефлексивной (на главной диагонали стоят нули) и симметричной (относительно главной диагонали).

Формат выходных данных

Выведите в выходной файл две строки. На первой выведите номера вершин, попавших во множество A , через пробел, а на второй — номера вершин, попавших во множество B , также через пробел. Номера вершин можно выводить в любом порядке.

Примеры

<code>mincut.in</code>	<code>mincut.out</code>
4	2
0111	1 3 4
1001	
1001	
1110	

Задача D. Улиточки

Имя входного файла: `snails.in`
Имя выходного файла: `snails.out`
Ограничение по времени: 2.5 секунды
Ограничение по памяти: 256 мегабайт

Две улиточки Маша и Петя сейчас находятся на лужайке с абрикосами и хотят добраться до своего домика. Лужайки пронумерованы числами от 1 до n и соединены дорожками (может быть несколько дорожек соединяющих две лужайки, могут быть дорожки, соединяющие лужайку с собой же). Ввиду соображений гигиены, если по дорожке проползла улиточка, то вторая по той же дорожке уже ползти не может. Помогите Пете и Маше добраться до домика.

Формат входных данных

В первой строке файла записаны четыре целых числа — n , m , a и h (количество лужаек, количество дорог, номер лужайки с абрикосами и номер домика).

В следующих m строках записаны пары чисел. Пара чисел (x, y) означает, что есть дорожка с лужайки x до лужайки y (из-за особенностей улиток и местности дорожки односторонние).

Ограничения: $2 \leq n \leq 10^5$, $0 \leq m \leq 10^5$, $a \neq h$

Формат выходных данных

Если существует решение, то выведите YES и на двух отдельных строчках сначала путь для Машеньки (т.к. дам нужно пропускать вперед), затем путь для Пети. Если решения не существует, выведите NO. Если решений несколько, выведите любое.

Примеры

<code>snails.in</code>	<code>snails.out</code>
3 3 1 3	YES
1 2	1 3
1 3	1 2 3
2 3	

Задача Е. Химия!!!

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

Вася и Сережа играют в следующую игру. В некоторых клетках клетчатого листка Сережа рисует один из символов 'H', 'O', 'N' или 'C', после чего Вася должен провести между некоторыми находящимися в соседних клетках символами линии так, чтобы получилось корректное изображение химической молекулы. К сожалению, Сережа любит рисовать много символов, и Вася не может сразу определить, возможно ли вообще нарисовать линии нужным способом. Помогите ему написать программу, которая даст ответ на этот вопрос.

В этой задаче проведенные между символами химических элементов линии будем считать корректным изображением молекулы, если они удовлетворяют следующим условиям:

- каждая линия соединяет символы, нарисованные в соседних (по стороне) клетках,
- между каждой парой символов проведено не более одной линии,
- от каждого элемента отходит ровно столько линий, какова валентность этого элемента (1 для H, 2 для O, 3 для N, 4 для C),
- пустые клетки ни с чем не соединены, и
- хотя бы в одной клетке нарисован какой-то символ.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m ($1 \leq n, m \leq 50$) — размеры листочка, на котором рисует Сережа. Далее следуют n строк по m символов в каждой, задающих конфигурацию химических элементов, которую нарисовал Сережа; пустые клетки задаются символом '.'.

Формат выходных данных

В выходной файл выведите одно слово: 'Valid', если линии провести требуемым образом можно, и 'Invalid', если нельзя.

Примеры

<code>molecule.in</code>	<code>molecule.out</code>
3 4 HOH. NSOH OO..	Valid
3 4 HOH. NSOH OONH	Invalid

Задача F. Perspective

Имя входного файла: `perspective.in`
Имя выходного файла: `perspective.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Баскетбол. NBA. Несколько команд играют турнир. Команды разбиты на дивизионы. Есть игры внутри дивизиона и между командами из разных дивизионов. Ничей не бывают, в каждой игре кто-то выигрывает, кто-то проигрывает. У вас есть любимая команда. Вы знаете всё про дивизион, в котором эта команда играет. Какие-то игры уже сыграны. Про каждую команду дивизиона вам известно, сколько побед уже одержала эта команда и сколько игр ещё предстоит ей сыграть (включая и игры внутри дивизиона, и игры вне дивизиона). Про каждую пару команд дивизиона известно, сколько ещё игр им предстоит сыграть друг с другом. Определите, могут ли все оставшиеся игры быть сыграны так, чтобы в результате у вашей любимой команды было побед не меньше чем у любой другой в её дивизионе?

Формат входных данных

На первой строке число команд в дивизионе n ($2 \leq n \leq 20$). Ваша любимая команда имеет номер 1. Следующая строка содержит n чисел, i -е число обозначает количество побед у i -й команды. Следующая строка содержит n чисел, i -е число обозначает количество предстоящих игр у i -й команды. Далее n строк содержат по n чисел, j -е число на i -й строке обозначает, количество предстоящих матчей между i -й и j -й командой дивизиона. Гарантируется, что матрица симметрична, и на диагонали стоят нули. Все числа целые, неотрицательные, до 10 000.

Формат выходных данных

Выведите YES или NO.

Примеры

<code>perspective.in</code>	<code>perspective.out</code>
3 1 2 2 1 1 1 0 0 0 0 0 0 0 0 0	YES
3 1 2 2 1 1 1 0 0 0 0 0 1 0 1 0	NO

Задача G. Чокнутый профессор

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

В Университете города М. проводят эксперимент. Преподаватели сами решают, что они будут читать в рамках того или иного курса. И вот преподаватель математического анализа (в простонародье — матана) оценил по некоторым критериям все известные ему темы в данном курсе. В результате этой ревизии каждой теме сопоставлено некоторое целое число (возможно, отрицательное) — полезность данной темы. Профессор хочет максимизировать суммарную полезность прочитанных им тем, но не все так просто. Для того что бы студенты поняли некоторые темы, необходимо, чтобы были прочитаны так же некоторые другие темы, так как некоторые доказательства базируются на фактах из других тем. Однако если существует цикл из зависимостей тем, то их все можно прочитать, и на качестве понимания материала студентами это не скажется.

Вас попросили составить список тем, которые профессор должен прочитать, таким образом, чтобы студенты все поняли, и суммарная полезность курса была максимальна.

Формат входных данных

Первая строка входного файла содержит одно число — N ($1 \leq N \leq 200$). Вторая строка содержит N целых чисел, не превосходящих по модулю 1000 — полезности каждой темы. Далее следуют N строк с описанием зависимостей тем. Каждое описание начинается количеством тем, которые необходимо понять для понимания данной темы. Потом следуют номера этих тем, разделенные пробелами.

Формат выходных данных

Выведите единственное число — максимально возможную суммарную полезность прочитанного материала.

Примеры

<code>matan.in</code>	<code>matan.out</code>
4 -1 1 -2 2 0 1 1 2 4 2 1 1	2
3 2 -1 -2 2 2 3 0 0	0