# Задача A. 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 входного файла

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

В первой строке каждого набора задаются числа $0 \le m \le 150$ и $0 \le n \le 150$. 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 выходного файла

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, разделённые одним пробелом. Следующие две строки должны содержать номера приглашённых юношей и приглашённых девушек соответственно. Числа в каждой из этих двух строк разделяются ровно одним пробелом и выводятся в порядке возрастания. Если максимальных наборов несколько, то выведите любой из них.

Разделяйте вывод для разных наборов входных данных одной пустой строкой.

## Пример

| birthday.in | birthday.out |
|---|---|
| 2<br>2 2<br>1 2 0<br>1 2 0<br>3 2<br>1 2 0<br>2 0<br>1 2 0 | 4<br>2 2<br>1 2<br>1 2<br><br>4<br>2 2<br>1 3<br>1 2 |

# Задача B. Максимальный поток

| | |
|---|---|
| Имя входного файла: | `flow2.in` |
| Имя выходного файла: | `flow2.out` |
| Ограничение по времени: | 0.5 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 входного файла

Первая строка входного файла содержит $n$ и $m$ — число вершин и ребер в графе ($2 \le n \le 500$, $1 \le m \le 10\,000$). 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 выходного файла

Выведите величину максимального потока между вершинами 1 и $n$.

## Примеры

| flow2.in | flow2.out |
|---|---|
| 4 5<br>1 2 1<br>1 3 2<br>3 2 1<br>2 4 2<br>3 4 1 | 3 |

# Задача C. Охлаждение реактора

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

Известная террористическая группа под руководством знаменитого террориста Бен Гадена решила построить атомный реактор для получения оружейного плутония. Вам, как компьютерному гению этой группы, поручили разработать систему охлаждения реактора.

Система охлаждения реактора представляет собой набор труб, соединяющих узлы. По трубам течет жидкость, причем для каждой трубы строго определено направление, в котором она должна по ней течь. Узлы системы охлаждения занумерованы от 1 до $N$. Система охлаждения должна быть спроектирована таким образом, чтобы для каждого узла за единицу времени количество жидкости, втекающей в узел, было равно количеству жидкости, вытекающей из узла. То есть если из $i$-го узла в $j$-ый течет $f_{ij}$ единиц жидкости за единицу времени (если из $i$ в $j$ нет трубы, то положим $f_{ij} = 0$), то для каждого узла $i$ должно выполняться

$$\sum_{j=1}^{N} f_{ij} = \sum_{j=1}^{N} f_{ji}$$

У каждой трубы имеется пропускная способность $c_{ij}$. Кроме того, для обеспечения достаточного охлаждения требуется, чтобы по трубе протекало не менее $l_{ij}$ единиц жидкости за единицу времени. То есть для трубы, ведущей из $i$-го узла в $j$-ый должно выполняться $l_{ij} \le f_{ij} \le c_{ij}$.

Вам дано описание системы охлаждения, выясните, каким образом можно пустить жидкость по трубам, чтобы выполнялись все указанные условия.

## Формат входного файла

Первая строка входного файла содержит числа $N$ и $M$ — количество узлов и труб ($1 \le N \le 200$). Следующие $M$ строк содержат описание труб. Каждая строка содержит четыре целых числа $i$, $j$, $l_{ij}$ и $c_{ij}$. Любые два узла соединены не более чем одной трубой, если есть труба из $i$ в $j$, то нет трубы из $j$ в $i$, никакой узел не соединен трубой сам с собой, $0 \le l_{ij} \le c_{ij} \le 10^5$.

## Формат выходного файла

Если решение существует, выведите на первой строке выходного файла слово `YES`. Затем выведите $M$ чисел — количество жидкости, которое должно течь по трубам, числа должны быть выведены в том порядке, в котором трубы заданы во входном файле. Если решения не существует, выведите `NO`.

## Пример

| cooling.in | cooling.out |
|---|---|
| 4 6<br>1 2 1 2<br>2 3 1 2<br>3 4 1 2<br>4 1 1 2<br>1 3 1 2<br>4 2 1 2 | NO |
| 4 6<br>1 2 1 3<br>2 3 1 3<br>3 4 1 3<br>4 1 1 3<br>1 3 1 3<br>4 2 1 3 | YES<br>1<br>2<br>3<br>2<br>1<br>1 |

# Задача D. Сетевые войны

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

Компьютерная сеть Байтландии состоит из $n$ компьютеров, соединенных $m$ оптоволоконными кабелями. Каждый кабель соединяет какие-то два компьютера и может передавать данные в обоих направлениях. Два компьютера сети особенно важны — один имеет выход в интернет, а другой соединен с президентской резиденцией.

Компьютер, соединенный с президентской резиденцией, имеет номер 1, а компьютер, имеющий выход в интернет, — номер $n$.

Недавно компания *Max Traffic* решила взять под контроль некоторые соединения, такие чтобы иметь возможность просматривать данные, передаваемые пользователями из президентской резиденции. Конечно же *Max Traffic* хочет контролировать такой набор соединений, чтобы все данные, передаваемые из интернета в президентскую резиденцию, проходили хотя бы через одно из контролируемых соединений.

Чтобы притворить свой план в жизнь, компании необходимо купить соответствующие соединения у их текущих владельцев. Каждое соединение имеет определенную цену. Поскольку основная специализация компании не шпионаж, а предоставление интернет услуг, руководство компании хочет, чтобы средняя цена покупки была минимальна. Если компания покупает $k$ соединений суммарной стоимостью $c$, то требуется минимизировать значение $c/k$.

### Формат входного файла

Первая строка входного файла содержит целые числа $n$ ($2 \le n \le 100$) и $m$ ($1 \le m \le 400$). Последующие $m$ строк содержат описания соединений — каждый кабель описывается тремя целыми числами: номерами компьтеров, которые он соединяет, и стоимостью покупки. Стоимость каждого соединения положительна и не превосходит $10^7$.

Любые два компьютера соединены не более, чем одним кабелем. Никакой компьютер не соединен сам с собой. Гарантируется, что сеть связна, т.е. данные можно передать от любого компьютера до любого другого по проводам сети.

### Формат выходного файла

На первой строке выведите число соединений, которое надо купить. Во второй строке выведите номера этих соединений. Соединения нумеруются начиная с 1 в том же порядке, в каком они заданы во входном файле.

### Пример

| network.in | network.out |
|---|---|
| 6 8<br>1 2 3<br>1 3 3<br>2 4 2<br>2 5 2<br>3 4 2<br>3 5 2<br>5 6 3<br>4 6 3 | 4<br>3 4 5 6 |
| 4 5<br>1 2 2<br>1 3 2<br>2 3 1<br>2 4 2<br>3 4 2 | 3<br>1 2 3 |

## Задача E. Craftsman

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

Takeshi, a famous craftsman, accepts many offers from all over Japan. However, the tools which he is using now has become already too old. So he is planning to buy new tools and to replace the old ones before next use of the tools. Some offers may incur him monetary cost, if the offer requires the tools to be replaced. Thus, it is not necessarily best to accept all the orders he has received. Now, you are one of his disciples. Your task is to calculate the set of orders to be accepted, that maximizes his earning for a given list of orders and prices of tools. His earning may shift up and down due to sale income and replacement cost. He always purchases tools from his friends shop. The shop discounts prices for some pairs of items when the pair is purchased at the same time. You have to take the discount into account. The total price to pay may be not equal to the simple sum of individual prices. You may assume that all the tools at the shop are tough enough. Takeshi can complete all orders with replaced tools at this time. Thus you have to buy at most one tool for each kind of tool.

### Формат входного файла

The input conforms to the following format:

$N\ M\ P$
$X_1 K_1 I_{1,1} \cdots I_{1,K_1}$
$\cdots$
$X_N K_N I_{N,1} \cdots I_{N,K_N}$
$Y_1$
$\cdots$
$Y_M$
$J_{1,1} J_{1,2} D_1$
$\cdots$
$J_{P,1} J_{P,2} D_P$

where $N$, $M$, $P$ are the numbers of orders, tools sold in the shop and pairs of discountable items, respectively. The following $N$ lines specify the details of orders. $X_i$ is an integer indicating the compensation for the $i$-th order, and $K_i$ is the number of tools required to complete the order. The remaining part of each line describes the tools required for completing the order. Tools are specified by integers from 1 through $M$. The next $M$ lines are the price list at the shop of Takeshis friend. An integer $Y_i$ represents the price of the $i$-th tool. The last $P$ lines of each test case

represent the pairs of items to be discounted. When Takeshi buys the $J_{i,1}$-th and the $J_{i,2}$-th tool at the same time, he has to pay only $D_i$ yen, instead of the sum of their individual prices. It is guaranteed that no tool appears more than once in the discount list, and that $max(Y_i, Y_j) < D_{i,j} < Y_i + Y_j$ for every discount prices, where $D_{i,j}$ is the discount price of $i$-th and $j$-th tools bought at the same time. Also it is guaranteed that $1 \leq N \leq 100, 2 \leq M \leq 100, 1 \leq Ki \leq 100, 1 \leq P \leq M/2$ and $1 \leq X_i, Y_i \leq 1000$.

### Формат выходного файла

Output the maximum possible earning of Takeshi to the standard output.

### Пример

| craftsman.in | craftsman.out |
|---|---|
| 3 4 2<br>100 2 1 2<br>100 1 3<br>100 1 4<br>20<br>20<br>50<br>150<br>1 2 30<br>3 4 180 | 120 |
| 1 2 1<br>100 1 2<br>20<br>40<br>1 2 51 | 60 |

## Задача F. Просто поток

| | |
|---|---|
| Имя входного файла: | flow.in |
| Имя выходного файла: | flow.out |
| Ограничение по времени: | 5 секунд |
| Ограничение по памяти: | 64 мегабайта |

Дана система из узлов и труб, по которым может течь вода. Для каждой трубы известна наибольшая скорость, с которой вода может протекать через нее. Известно, что вода течет по трубам таким образом, что за единицу времени в каждый узел (за исключением двух — источника и стока) втекает ровно столько воды, сколько из него вытекает. Более того, известно, что для любой пары узлов (включая источник и сток) сумма скоростей течения воды вдоль любого пути, их соединяющего, постоянна для данной пары узлов. Сумма берется таким образом, что если труба представлена в пути против направления движения воды в ней, то соответствующее слагаемое берется со знаком минус.

Ваша задача — найти наибольшее количество воды, которое за единицу времени может протекать между источником и стоком, а также скорость течения воды по каждой из труб.

Трубы являются двусторонними, то есть вода в них может течь в любом направлении. Между любой парой узлов может быть более одной трубы.

### Формат входного файла

В первой строке записано натуральное число $N$ — количество узлов в системе ($2 \leq N \leq 100$). Известно, что источник имеет номер 1, а сток номер $N$. Во второй строке записано натуральное $M$ ($1 \leq M \leq 5000$) — количество труб в системе. Далее в $M$ строках идет описание труб. Каждая труба задается тройкой целых чисел $A_i$, $B_i$, $C_i$, где $A_i$, $B_i$ — номера узлов, которые соединяет данная труба, а $C_i$ ($0 \leq C_i \leq 10000$) — наибольшая допустимая скорость течения воды через данную трубу.

### Формат выходного файла

В первой строке выведите наибольшее количество воды, которое протекает между источником и стоком за единицу времени. Далее выведите $M$ строк, в каждой из которых выведите скорость течения воды по соответствующей трубе. Если направление не совпадает с порядком узлов, заданным во входных данных, то выводите скорость со знаком минус. Числа выводите с точностью $10^{-3}$.

### Пример

| flow.in | flow.out |
|---|---|
| 2<br>1<br>1 2 1 | 1.0000000<br>1.0000000 |

# Часть 2: к 1-му апреля 2011-го года

## Задача G. Улиточки

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

Две улиточки Маша и Петя сейчас находятся в 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). В виду 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 входного файла

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: $2 \le n \le 10^5, 0 \le m \le 10^5, s \ne t$.

### Формат выходного файла

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

### Пример

| snails.in | snails.out |
|---|---|
| 3 3 1 3<br>1 2<br>1 3<br>2 3 | YES<br>1 3<br>1 2 3 |

## Задача H. Molecule. Химия!!!

| | |
|---|---|
| Имя входного файла: | `molecule.in` |
| Имя выходного файла: | `molecule.out` |
| Ограничение по времени: | 1 секунда |
| 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 входного файла

Первая строка входного файла содержит два натуральных числа $n$ и $m$ ($1 \le n, m \le 50$) — 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 выходного файла

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

### Пример

| molecule.in | molecule.out |
|---|---|
| 3 4<br>HOH.<br>NCOH<br>OO.. | Valid |
| 3 4<br>HOH.<br>NCOH<br>OONH | Invalid |

## Задача I. Великая стена

| | |
|---|---|
| Имя входного файла: | wall.in |
| Имя выходного файла: | wall.out |
| Ограничение по времени: | 4 секунды |
| Ограничение по памяти: | 64 мегабайта |

У короля Людовика двое сыновей. Они ненавидят друг друга, и король боится, что после его смерти страна будет уничтожена страшными войнами. Поэтому Людовик решил разделить свою страну на две части, в каждой из которых будет властвовать один из его сыновей. Он посадил их на трон в города $A$ и $B$, и хочет построить минимально возможное количество фрагментов стены таким образом, чтобы не существовало пути из города $A$ в город $B$.

Страну, в которой властвует Людовик, можно упрощенно представить в виде прямоугольника $m \times n$. В некоторых клетках этого прямоугольника расположены горы, по остальным же можно свободно перемещаться. Кроме этого, ландшафт в некоторых клетках удобен для строительства стены, в остальных же строительство невозможно.

При поездках по стране можно перемещаться из клетки в соседнюю по стороне, только если ни одна из этих клеток не содержит горы или построенного фрагмента стены.

### Формат входного файла

В первой строке входного файла содержатся числа $m$ и $n$ ($1 \le m, n \le 50$). Во второй строке заданы числа $k$ и $l$, где $0 \le k, l, k + l \le mn - 2$, $k$ — количество клеток, на которых расположены горы, а $l$ — количество клеток, на которых можно строить стену. Естественно, что на горах строить стену нельзя. Следующие $k$ строк содержат координаты клеток с горами $x_i$ и $y_i$, а за ними следуют $l$ строк, содержащие координаты клеток, на которых можно построить стену — $x_j$ и $y_j$. Последние две строки содержат координаты городов $A$ ($x_A$ и $y_A$) и $B$ ($x_B$ и $y_B$) соответственно. Среди клеток, описанных в этих $k + l + 2$ строках, нет двух совпадающих. Гарантируется, что $1 \le x_i, x_j, x_A, x_B \le m$ и $1 \le y_i, y_j, y_A, y_B \le n$.

### Формат выходного файла

В первой строке выходного файла должно быть выведено минимальное количество фрагментов стены $F$, которые необходимо построить. В последующих $F$ строках необходимо вывести один из возможных вариантов застройки.

Если невозможно произвести требуемую застройку, то необходимо вывести в выходной файл единственное число $-1$.

### Пример

| wall.in | wall.out |
|---|---|
| 5 5<br>3 8<br>3 2<br>2 4<br>3 4<br>3 1<br>1 3<br>2 3<br>3 3<br>4 3<br>5 3<br>1 4<br>1 5<br>2 1<br>5 5 | 3<br>3 1<br>1 3<br>3 3 |

## Задача J. Ёжики

| | |
|---|---|
| Имя входного файла: | hedgehog.in |
| Имя выходного файла: | hedgehog.out |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 256 мегабайт |

К

Дан неориентированный граф без петель и кратных ребер, требуется найти в нем:

1. Размер минимального контролирующего множества вершин.

2. Размер максимального независимого множества вершин.
3. Размер минимального глобального разреза.

**Def:** *контролирующее множество* — множество вершин, такое, что у каждого ребра хотя бы один из двух концов лежит в множестве.

**Def:** *независимое множество* — множество вершин, такое, что никакие две вершины множества не соединены ребром.

**Def:** *глобальный разрез* — разбиение множества всех вершин $V$ на непустые $S$ и $T$ ($S \cup T = V, S \cap T = \varnothing$). Величиной разреза называется количество ребер между $S$ и $T$.

### Формат входного файла

В первой строке файла записаны два числа — $n$, $m$ (количество вершин и количество ребер).

В следующих $m$ строках записаны пары чисел. Пара чисел $xy$ означает, что есть ребро между вершинами $x$ и $y$.

Ограничения: $2 \le n \le 15$.

### Формат выходного файла

Выведите 3 числа — ответы для трех задач.

### Пример

| hedgehog.in | hedgehog.out |
|---|---|
| 4 4<br>1 2<br>2 3<br>3 1<br>3 4 | 1 2 1 |

## Задача K. Глобальный разрез

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

Дан взвешенный неориентированный граф. Нужно удалить минимальное по сумме весов множество ребер так, чтобы граф стал несвязным. После того, как граф становится несвязным, множество вершин распадается на два непустых множества $S$ и $T$. Зная одно из двух множеств можно восстановить второе и удаленные ребра.

### Формат входного файла

Количество вершин и ребер в графе — $1 \le N \le 500, 1 \le M \le N * (N-1)/2$.

Далее $M$ строк, в каждой 3 числа — $1 \le a, b \le N$ и $0 \le w \le 10^9$ (вершины, которые соединяет очередное ребро, и вес ребра).

### Формат выходного файла

Минимальный суммарный вес ребер, которых нужно удалить.

Далее размер множества $S$ и собственно вершины из $S$.

### Пример

| stor.in | stor.out |
|---|---|
| 3 3<br>1 2 2<br>2 3 3<br>3 1 1 | 3<br>2<br>2 3 |
| 6 7<br>1 2 10<br>2 3 10<br>3 1 10<br>4 5 10<br>5 6 10<br>6 4 10<br>2 5 1 | 1<br>3<br>4 5 6 |