Задача A. Kingdoms

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 mebibytes

Во времена экономического благополучия n соседних королевств секретно занимали друг у друга значительные суммы. Однако сайт «Wikipeaks» не так давно опубликовал общие суммы задолженностей. После чего кризис стал уже неминуем. Короли обратились в Международный Валютный Фонд (МВФ), чтобы тот помог решить вопрос с финансовым оздоровлением.

Для каждой пары (A, B) королевств известно количество золота d_{AB}, которое королевство A должно королевству B или наоборот, то есть $d_{BA} = -d_{AB}$. МВФ выбирает одно из королевств, сумма долгов которого является положительной, и объявляет в этом королевстве дефолт — выбранное королевство выходит изо всех финансовых договоров (причём как тех, в которых оно выступало заёмщиком, так и тех, в которых оно выступало заимодавцем).

После этого рассматривается уже изменившаяся ситуация, снова выбирается для дефолта одно из королевств-должников и так далее, пока хотя бы у какого-нибудь из оставшихся королевств сумма долгов превышает 0.

В зависимости от порядка выбора королевств для дефолта могут осуществиться различные сценарии: например, может оказаться, что только одно королевство не пройдёт процедуры дефолта. Для кажжого королевства определите, может ли оно остаться единственным, в котором не объявлен дефолт?

Формат входных данных

В первой строке входного файла задано целое число $T \le 10^4$ — количество тестовых примеров. Далее следуют описания тестовых примеров.

Каждый тестовый пример начинается со строки, содержащей n — количество королевств, $1 \le n \le 20$. Королевства занумерованы, начиная с единицы.

Далее заданы n строк, каждая из которых содержит n чисел, разделённых пробелами. j-е числов в i-й строке задаёт число d_{ij} — сумму, которую i-е королевство должно j-му. Гарантируется, что $d_{ii} = 0$ и $d_{ij} = -d_{ji}$ для каждого $1 \le i, j \le n$. Также гарантируется, что $|d_{ij}| \le 10^6$ для всех пар i и j.

Гарантируется, что сумма всех n в тестовом примере не превосходит $5 \cdot 10^4$.

Формат выходных данных

Для каждого тестового примера в порядке их следования во входном файле выведите строку, содержащую номера королевств, каждое из которых при определённом сценарии может остаться единственным, не прошедшем через процедуру дефолта. В случае, если таких королевств не существует, строка должна содержать единственное число 0.

Пример

standard input	standard output
1 3 0 -3 1 3 0 -2 -1 2 0	1 3

Задача B. Who wants to live forever? (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 mebibytes

«Цифровая физика» — это популярный в Байтландии набор идей и гипотез, работающих с концепцией вычислимого мироздания. Согласно этой концепции, мироздание — это большая программа, которая запущена на машине Тьюринга. Таким образом, ответ на вопрос о том, будет ли существовать данная Вселенная бесконечно или же погибнет, становится вычислимым.

В качестве простейшего примера универсума рассматривается *уноверсум* — вселенная, моделируемая n последовательными битами, занумерованными с единицы. В момент «большого взрыва» биты были инициализированы некоторым образом. Далее уноверсум развивается пошагово следующим образом. Значение i-го бита на $j+1$-м шаге равно результату операции «исключающее или» над значениями $i-1$-го и $i+1$-го бита на j-м шаге (0-й и $n+1$-й бит считаются равными нулю). Каждый следующий шаг начинается после завершения предыдущего. В случае, если значения всех бит уноверсума становятся равными нулю, считается, что уноверсум погибает.

По заданному значению битов в момент «большого взрыва» определите, будет ли уноверсум жить вечно или в какой-то момент погибнет.

Формат входных данных

В первой строке входного файла задано целое число $T \leq 65535$ — количество тестовых примеров. Далее следуют описания тестовых примеров.

Каждый тестовый пример состоит из не менее, чем одного, и не более, чем $2 \cdot 10^5$ символов ‘0’ или ‘1’.

Гарантируется, что размер входного файла не превосходит 10 мебибайт.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите “LIVES”, если соответствующий уноверсум будет жить вечно, и “DIES” в противном случае.

Пример

standard input	standard output
3 01 0010100 11011	LIVES DIES LIVES

Замечание

Уноверсум из первого примера будет циклично менять состояние с 01 на 10. Второй погибнет после трёх шагов (0010100 0100010 1010101 0000000), состояние третьего вообще не будет меняться.

Задача C. Chemist's vows

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 mebibytes

Химик Клара Каннибал-Щукина дала своеобразный обет молчания — она решила произносить только те слова, которые можно составить из элементов таблицы Менделеева. Например, она может сказать "`I Am CLaRa`" (так как I — знак йода, Am — для америция, C — для углерода, La —- для лантана и Ra — для радия), или же "`InTeRnAtIONAl`", но не может сказать, например, "`collegiate`", "`programming`" и "`contest`".

Учёные решили составить словарь используемых Каннибал-Щукиной слов. По заданному слову определите, сможет ли Клара произнести его. При этом внутренняя капитализация в слове может быть произвольной. Таблица Менделеева в том виде, в котором она была использована Кларой, дана ниже.

```
H                                                       He

Li  Be                                  B   C   N   O   F   Ne

Na  Mg                                  Al  Si  P   S   Cl  Ar

K   Ca  Sc  Ti  V   Cr  Mn  Fe  Co  Ni  Cu  Zn  Ga  Ge  As  Se  Br  Kr

Rb  Sr  Y   Zr  Nb  Mo  Tc  Ru  Rh  Pd  Ag  Cd  In  Sn  Sb  Te  I   Xe

Cs  Ba  *   Hf  Ta  W   Re  Os  Ir  Pt  Au  Hg  Tl  Pb  Bi  Po  At  Rn

Fr  Ra  **  Rf  Db  Sg  Bh  Hs  Mt  Ds  Rg  Cn      Fl      Lv


*  La  Ce  Pr  Nd  Pm  Sm  Eu  Gd  Tb  Dy  Ho  Er  Tm  Yb  Lu
** Ac  Th  Pa  U   Np  Pu  Am  Cm  Bk  Cf  Es  Fm  Md  No  Lr
```

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров $T \leq 10^4$. Далее следуют описания тестовых примеров.

Каждый тестовый пример содержит единственное слово, составленное из строчных латинских букв. Слово является непустым, а его длина не превосходит $5 \cdot 10^4$.

Гарантируется, что размер входного файла не превышает $6 \cdot 10^5$ байт.

Формат выходных данных

Для каждого тестового примера в порядке их следования во входном файле выведите в отдельной строке "`YES`", если Клара может произнести соответствующее слово, и "`NO`" в противном случае.

Пример

standard input	standard output
4 international collegiate programming contest	YES NO NO NO

Замечание

По адресу http://acm.math.spbu.ru:17249/files/opencup/oc12/gp6/c.tex можно найти TeX-овский исходник данной задачи.

Задача D. Non-boring sequences

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 mebibytes

Назовём последовательность A *не занудной*, если для произвольной подпоследовательности B, составленной из идущих подряд элементов A, найдётся «уникальный» элемент, то есть элемент, который встречается среди элементов подпоследолвательности B ровно один раз.

По заданной последовательности установите, является ли она не занудной.

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров T. Далее следуют описания тестовых примеров.

Каждый тестовый пример начинается с целого числа n ($1 \le n \le 2 \cdot 10^5$) — длины последовательности. В следующей строке перечислены в порядке возрастания индексов элементы последовательности — n целых неотрицательных чисел, меньших 10^9.

Сумма всех n во входном файле не превышает $1.2 \cdot 10^6$.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите "`non-boring`" в случае, если последовательность является не занудной и "`boring`" в противном случае.

Пример

standard input	standard output
4 5 1 2 3 4 5 5 1 1 1 1 1 5 1 2 3 2 1 5 1 1 2 1 1	non-boring boring non-boring boring

Задача E. Word equations

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Задан текст T и шаблон P. Требуется выяснить, можно ли удалить несколько символов из текста T так, чтобы в результате получился шаблон P. Например, из слова "programming" можно таким образом получить слова "pong", "program", "roaming", но не слово "map", так как буквы должны оставаться в том же порядке. При этом слова состоят только из строчных латинских букв.

Текст T, однако, задан системой уравнений. В уравнениях используются переменные (каждая переменная представляет собой слово, состоящее только из заглавных латинских букв). Каждая переменная обозначает некоторое слово, состоящее из строчных латинских букв (то есть слово над алфавитом $\{a, \ldots, z\}$. Уравнения принадлежат к одному из двух видов:

$$A = a\ word\ over \{a, \ldots, z\}$$

то есть переменной ставится в соответствие слово, или

$$A = B + C$$

В последнем случае A, B, C — переменные (при этом переменные B и C могут совпадать), а знак `+` обозначает конкатенацию слов (как `+`, так и `=` отделяются от имён переменных пробелами). Гаранитруется, что система

- однозначная— для заданной переменной A существует ровно одно уравнение, в котором A стоит в левой части, и
- ациклическая — если вы начинаете с произвольной переменной A и проводите подстановки в соответствии с уравнениями, вы не получится снова выражения, содержащего A.

Тем самым система всегда имеет единственное решение. Например, для системы

```
START = FIRST + SECND
FIRST = D + E
SECND = F + E
D = good
E = times
F = bad
```

значением переменной START является слово "`goodtimesbadtimes`".

Вам задано слово P, система уравнений и некоторая переменная S, входящая в эту систему. Требуется выяснить, можно ли получить слово P путём удаления нескольких букв из слова, являющегося значенем переменной S.

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров $T \leq 500$. Далее следуют описания тестовых примеров.

Каждый тестовый пример начинается со строки, содержащей целое число k ($1 \leq k \leq 500$) — количество уравнений. В последующих k строках заданы уравнения. Каждое из них принадлежит к одному из описанных в условии задачи видов, при этом слова, знаки '+' и '='

отделяются друг от друга ровно одним пробелом. Слова и значения переменных имеют ненулевую длину, не превосходящую 5. Далее следуют две строки, первая из которых содержит имя переменной, присутствующее в системе, а вторая содержит непустое слово длиной не более 2000, состоящее из строчных латинских букв — соответственно переменная, из значения которой должно быть получено путём удаления некоторого (возможно, нулевого) числа букв слово, и само это слово.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите "YES", если слово можно найти в значении соответствующей переменной, и "NO" в противном случае.

Пример

standard input	standard output
1 6 START = FIRST + SECND FIRST = D + E SECND = F + E D = good E = times F = bad START debate	YES

Задача F. Farm and factory (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	15 seconds
Ограничение по памяти:	256 mebibytes

Во время правления короля Байтоломью в Байтландии концентрация промышленного и сельскохозяйственного производства достигла невероятных масштабов. Все жители, кроме короля, работали либо на одной большой ферме, либо на одной большой фабрике. Ферма и фабрика были расположены в двух разных городах. Так что каждое утро жители отправлялись в соответствующие города на работу и службу, что создавало невероятные для XIX века заторы.

В те годы дорожная сеть Байтландии состояла из дорог с двусторонним движением, каждая из которых соединяет два различных города. Дороги не пересекаются вне городов. При этом между двумя городами могло быть построено более одной прямой дороги. Разумеется, и до фермы, и до фабрики можно добраться из любого города.

Когда Байтоломью доложили о заторах, король издал указ, устанавливающий плату за проезд по каждой дороге. Теперь каждый житель при проезде по некоторой дороге вносил в королевскую казну плату; для каждой дороги тариф был установлен отдельно. Указ был одобрен парламентом и вступил в силу.

В результате все жители стали для поездки на работу выбирать наиболее экономичные пути, а казна обогатилась до такой степени, что парламент одобрил давний королевский проект о строительстве новой столицы с роскошным королевским дворцом. Новую столицу надо соединить с несколькими существующими городами прямыми дорогами так, чтобы из столицы можно было добраться по сети дорог в любой город королевства.

После строительства дорог возникла проблема следующего свойства. Если король установит слишком низкую плату за проезд по новым дорогам, то столица будет забита транспортом жителей страны, едущих на работу и с работы (так как через столицу будет проходить самый экономичный маршрут для некоторых городов), если же установит слишком высокую, то пострадает его личный бюджет (законы Байтландии не освобождают короля от налогов).

Так что король хочет установить плату за проезд по ведущим от вновь построенной столицы дорогам так, чтобы для каждого города, отличного от столицы, существовал более дешёвый путь на фабрику и ферму соответственно, не проходящий через столицу (в том числе и для городов, где находятся соответственно ферма и фабрика). С другой стороны, король хочет минимизировать среднюю стоимость проезда от новой столицы до произвольного города в королевстве.

Ваша задача — определить минимальное значение средней стоимости проезда от новой столицы до произвольного города королевства. Тарифы на проезд по новым дорогам не могут быть отрицательными (но не обязаны быть целыми).

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров $T \le 1.1 \cdot 10^4$. Далее следуют описания тестовых примеров.

Первая строка каждого тестового примера содержит два целых числа n, m ($2 \le n \le 10^5,\ 1 \le m \le 3 \cdot 10^5$) — количество байтландских городов в правление Байтоломью и количество дорог соответственно. В последующих m строках заданы дороги. Каждая дорога задаётся тремя целыми числами u, v и c ($1 \le u, v \le n, u \ne v,\ 0 \le c \le 10^6$) — номера городов, соединённых данной дорогой, и плата за проезд по этой дороге. При этом может существовать

несколько дорог, соединяющих одни и те же города.

Ферма и фабрика расположены в городах с номерами 1 и 2 соответственно.

Сумма всех n для одного входного файла не превышает $6 \cdot 10^5$.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите минимально возможное значение средней стоимости проезда от вновь возведённой столицы до каждого из городов королевства. Ответ выведите с абсолютной или относительной точностью не хуже 10^{-8}.

Пример

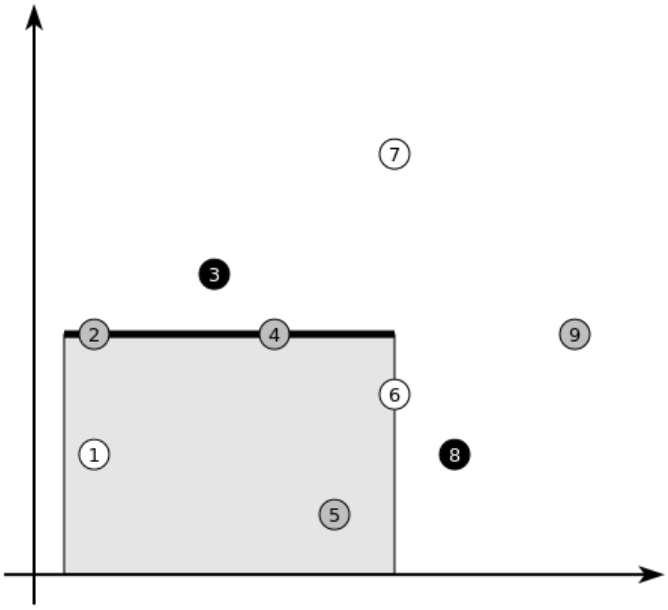
standard input	standard output
1 3 3 1 2 5 2 3 5 3 1 1	1.833333333333

Задача G. Jewel Heist (Division 1 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	15 seconds
Ограничение по памяти:	256 mebibytes

Ловкий вор Арсен Люпен собирается украсть драгоценные камни Злого Эрвина. На витрине ювелирной лавки Злого Эрвина расположено n камней. Каждый драгоценный камень может быть одного из k попарно различных цветов. В данной задаче мы представим камни на витрине в виде n попарно различных точек координатной плоскости.

Для этой авантюры Арсен Люпен разработал хитрое устройство: механическую руку, которая может собрать часть драгоценных камней с витрины. Рука проходит по витрине только один раз и забирает все камни, которые находятся на заданном горизонтальном отрезке или ниже их. Казалось бы, таким образом легко можно забрать все камни... но Арсен Люпен знает, что Эрвин оборудовал витрину хитрой системой сигнализации, которая сработает, если с витрины исчезнет набор камней, содержащий все k цветов. Таким образом, Арсен хочет украсть максимальное число камней так, чтобы среди них было не более $k-1$ камней попарно различных цветов.



The robotic hand grabs jewels 1,2,4,5 and 6, carefully omitting the black ones

Вычислите, какое максимальное количество камней Арсен Люпен сможет украсть.

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров $T \le 10^4$. Далее следуют описания тестовых примеров.

Первая строка каждого тестового примера содержит два целых числа: n ($2 \le n \le 2 \cdot 10^5$) и k ($2 \le k \le n$), обозначающие количество камней на витрине и количество различных цветов. Каждая из последующих n строк задаёт координаты и цвет одного из камней. j-я строка содержит три целых числа x_j, y_j, c_j ($1 \le x_j, y_j \le 10^9$, $1 \le c_j \le k$) — координаты (x_j, y_j) j-го камня и его цвет c_j.

Гарантируется, что на витрине находится как минимум один из камней каждого из k цветов.

Сумма всех n в одном входном файле не превышает 10^6.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите одно целое число — наибольшее количество камней, которое может украсть Арсен Люпен без срабатывания сигнализации.

Пример

standard input	standard output
1 10 3 1 2 3 2 1 1 2 4 2 3 5 3 4 4 2 5 1 2 6 3 1 6 7 1 7 2 3 9 4 2	5

Задача H. Darts

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Рассмотрим следующий вариант игры в дартс. Мишень представляет собой 10 концентрических кругов с радиусами 20, 40, 60, 80, 100, 120, 140, 160, 180, и 200 миллиметров и центром в начале координат. Каждый бросок оценивается в соответствии с тем, куда попал дротик. Бросок оценивается в p баллов ($p \in \{1, 2, \ldots, 10\}$), если минимальный круг, внутри или на границе которого находится точка попадания, имеет радиус $20 \cdot (11 - p)$. В случае, если бросок попадает вне мишени (то есть не попадает в наибольший круг), бросок оценивается в 0 баллов.

По заданным точкам попадания n бросков вычислите суммарное количество набранных баллов.

Формат входных данных

Первая строка входного файла содержит количество тестовых примеров $T \le 10^4$. Далее следуют описания тестовых примеров.

Каждый тестовый пример начинается со строки, содержащей n — количество бросков ($1 \le n \le 10^6$). Каждая из последующих n строк содержит по два целых числа x и y ($-200 \le x, y \le 200$) — координаты соответствующего попадания.

Сумма всех n во входном файле не превосходит $1.1 \cdot 10^6$.

Формат выходных данных

Для каждого тестового примера выведите в отдельной строке одно целое число — суммарное количество набранных за n бросков баллов.

Пример

standard input	standard output
1 5 32 -39 71 89 -60 80 0 0 196 89	29

Задача I. The Dragon and the knights

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	5 seconds
Ограничение по памяти:	256 mebibytes

Дракон Вавельского замка после конфликта с местной гильдией сапожников решил переместиться из Кракова в новое место. Изучив все возможные варианты, дракон выбрал Байтландское королевство.

По территории Байтландского королевства протекает n рек, каждая из которых течёт вдоль некоторой прямой. Реки могут быть параллельными. Никакие три реки не пересекаются в одной точке. Реки делят страну на несколько *дистриктов*. Королевство столь велико, что его территорию можно считать бесконечной плоскостью.

В Байтландии на момент появления дракона было m рыцарей. Каждый из рыцарей защищает тои дистрикт, в котором находится его замок, от внешней угрозы, так что дракон не поселится в дистрикте, в котором находится хотя бы один рыцарь.

У дракона есть карта королевства и координаты замков. Дракон просит Вас определить, все ли дистрикты королевства защищены, или же существует уязвимый дистрикт (в котором дракон и поселится).

Формат входных данных

В первой строке входного файла задано целое число $T \le 75$ — количество тестовых примеров. Далее задаются тестовые примеры.

Каждый тестовый пример начинается со строки, содержащей два целых числа — количество рек n ($1 \le n \le 100$) и количество рыцарей m ($1 \le m \le 5 \cdot 10^4$).

Последующие n строк задают реки. j-я из этих строк содержит три целых числа A_j, B_j, C_j, не превосходящие по абсолютному значению 10^4 — коэффициенты уравнения $A_j \cdot x + B_j \cdot y + C_j = 0$, задающего прямую, вдоль которой течёт река.

Далее следуют m строк, задающие координаты рыцарских замков: i-я из этих строк содержит два целых числа X_i и Y_i ($-10^9 \le X_i, Y_i \le 10^9$) — координаты замка i-го рыцаря. При этом ни один замок не может находиться в реке. Несколько рыцарей могут жить в одном замке (то есть для них обе координаты замка совпадают). Никакие две реки не текут по одной и той же прямой и никакие три реки не пересекаются в одной точке.

Сумма всех m и n в одном входном файле не превосходит $6 \cdot 10^5$.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите "`PROTECTED`", если все дистрикты защищены рыцарями, и "`VULNERABLE`" в противном случае.

Пример

standard input	standard output
2 3 7 0 1 0 1 0 0 1 1 -3 1 1 5 -1 3 2 2 -2 -2 6 -1 -2 -8 4 1 1 0 1 0 0 1	PROTECTED VULNERABLE

Задача J. Conservation

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	8 seconds
Ограничение по памяти:	256 mebibytes

Один из выдающихся шедевров байтландского изобразительного искусства — портрет дамы с компьютерной мышью — находится под угрозой и должен быть немедленно отправлен на реставрацию. Реставрационные работы будут проводиться двумя реставрационными лабораториями. Процесс реставрации состоит из нескольких видов работ; при этом для каждого типа работ однозначно определено, какая из лабораторий его будет производить.

Учитывая плачевное состояние картины, реставрвторы хотят минимизировать количество её перемещений между лабораториями. В идеальном варианте картина попадает в одну из лабораторий, там с ней проделываются все работы, которые должна производить первая лаборатория, затем картина переезжает в другую лабораторию для завершения процесса реставрации. Однако не всё так просто: для некоторых пар типов работ существуют зависимости: этап A обязан быть завершён ранее этапа B.

По заданной таблице зависимостей определите наименьшее возможное количество перемещений картины между лабораториями в процессе реставрации.

Формат входных данных

Первая строка входного файла содержит одно целое число T — количество тестовых примеров. Далее следуют описания тестовых примеров.

Первая строка каждоого тестового примера содержит два целых числа n и m, разделённые пробелами ($1 \le n \le 10^5$, $0 \le m \le 10^6$) — количество видов необходимых реставрационных работ и количество зависимостей между различными работами.

В следующей строке заданы n целых чисел, разделённых пробелами. i-е из них равно 1, если соответствующая работа выполняется первой лабораторией, и 2, если второй. Каждая из последующих m строк содержит два целых числа i, j ($1 \le i, j \le n$), и обозначает, что i-я работа должна быть завершена перед j-й.

Гарантируется, что существует такой порядок реставрационных работ, что все зависимости будут выполняться.

Сумма всех n во входном файле не превосходит $5 \cdot 10^5$. Сумма всех m во входном файле не превосходит $3 \cdot 10^6$.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите минимально возможное количество перемещений между лабораториями в процессе реставрации. Если вся работа производится в одной лаборатории, выведите 0.

Пример

standard input	standard output
1 5 6 1 2 1 2 1 1 2 1 3 2 4 3 4 2 5 3 5	2

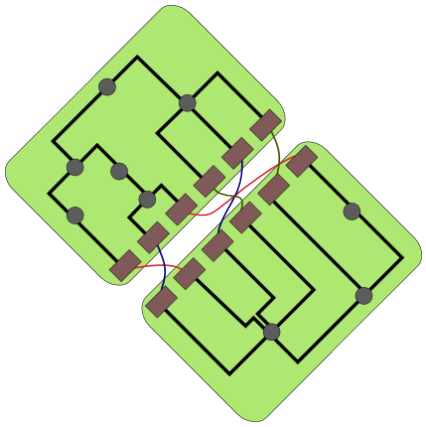
Задача K. Graphic Madness

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

На рынке видеоускорителей в Байтландии конкурируют два крупных производителя видеокарт: MAD и N-audia. Недавно они практически синхронно выпустили топовые видеокарты своей новой линейки. Каждую из карт можно представить как некоторое количество узлов, соединённых проводящими дорожками. Узлы подразделяются на два типа: сокеты и процессоры. Сеть проводящих дорожек удовлетворяет следующим условиям:

- Каждый сокет соединён ровно с одним процессором и не соединён ни с каким другим сокетом.
- Каждый процессор соединён как минимум с двумя другими узлами.
- Для любых двух узлов сети существует ровно один путь между ними по проводящим дорожкам. Иначе говоря, граф соединений между узлами является деревом.

Прочитав в спецификациях, что количество сокетов на карте от MAD совпадает с количеством сокетов на карте от N-audia, хакер Том провёл и описал на своём сайте интересный эксперимент. Том соединил с помощью внешних кабелей каждый сокет карты от MAD с сокетом карты от N-audia так, что разным сокетам соответствуют разные. Получилось что-то типа этого:



Для следующей статьи хакер Том обдумывает, каким образом получить максимальный эффект от подобного соединения. Он хочет рассчитать путь сигнала по кабелям и дорожкам, который бы проходил через каждый узел каждой карты ровно один раз, а также начинался бы и заканчивался в одном и том же узле одной и той же карты. По заданным схемам карт и соединения выясните, существует ли такой путь.

Формат входных данных

В первой строке входного файла содержится число тестовых примеров $T \le 35$. Далее задаются тестовые примеры.

В первой строке каждого тестового примера заданы три целых числа k, n, m ($2 \le k \le 1000$, $1 \le n, m \le 1000$) — количество сокетов на каждой карте, количество процессоров на карте MAD и количество процессоров на карте N-audia. Узлы на картах обозначаются следующим образом:

- сокеты на карте MAD: AS1, AS2, ..., ASk
- процессоры на карте MAD: AP1, AP2, ..., APn
- сокеты на карте N-audia: BS1, BS2, ..., BSk
- Процессоры на карте N-audia: BP1, BP2, ..., BPm

Последующие $n + k - 1$ строк содержат описание проводящих дорожек на карте MAD. Каждая из этих строк содержит названия двух различных узлов этой карты, соединённых напрямую дорожкой. Далее следует пустая строка, затем, в последующих $m + k - 1$ строках, аналогичным образом описаны проводящие дорожки на карте N-audia. Далее следует ещё одна пустая строка, после чего в k строках описываются добавленные Томом кабели. Каждая из этих строк содержит названия двух сокетов на различных картах, которые соединены напрямую внешним кабелем. Название каждого сокета встречается среди этих k строк ровно один раз. После каждого тестового примера, кроме последнего, также следует пустая строка.

Формат выходных данных

В выходной файл для каждого тестового примера выведите одну строку. В случае, если описанного в условии задачи пути не существует, выведите "NO". В противном случае выведите "YES", а далее, через пробел, выведите требуемый путь: названия $n+m+2k$ различных узлов, заданных в порядке прохождения через них сигнала. Каждые два соседних узла (а также первый или последний) должны быть соединены напрямую дорожкой или внешним кабелем.

Пример

standard input
1 2 1 11 AS1 AP1 AS2 AP1 BS1 BP1 BS2 BP11 BP1 BP2 BP2 BP3 BP3 BP4 BP4 BP5 BP5 BP6 BP6 BP7 BP7 BP8 BP8 BP9 BP9 BP10 BP10 BP11 AS1 BS2 BS1 AS2
standard output
YES BP11 BP10 BP9 BP8 BP7 BP6 BP5 BP4 BP3 BP2 BP1 BS1 AS2 AP1 AS1 BS2

Задача L. User name (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Операционная система PiOS генерирует логины по имени и фамилии в соответствии со следующими правилами:

1. Наибольшая длина логина — MAXLEN символов.
2. Первый символ логина совпадает с первой буквой имени, записанной как строчная буква.
3. Добавляется столько букв фамилии, сколько возможно без превышения MAXLEN. Все эти буквы конвертируются в строчные. Дефисы и апострофы при этом игнорируются.
4. Если выбранный после шагов $1-3$ логин уже существует, к полученному логину дописывается наименьшая цифра от 1 до 9 такая, что соответствующего логина ещё не существует. В случае, если логин первоначально имел длину MAXLEN, от него отбрасывается последняя буква.
5. Если выбранный после шагов $1-3$ логин и все 9 логинов, перебиравшихся на шаге 4, существуют, к полученному на шаге 3 логину дописывается наименьшее двузначное число от 10 до 99 такое, что соответствующего логина ещё не существует. В случае, если логин первоначально имел длину MAXLEN-1 или MAXLEN, от него отбрасываются одна или две, соответственно, последние буквы.
6. Гарантируется, что применение шагов $1-5$ всегда даёт возможность выбрать уникальный логин.

Вам заданы несколько имён и фамилий пользователей в порядке создания соответствующих пользователей. Требуется восстановить сгенерированные по соответствующим правилам логины.

Формат входных данных

Входной файл состоит из нескольких тестовых примеров. Первая строка каждого тестового примера содержит два целых положительных числа — количество пользователей U ($1 \le U \le 200$) и значение MAXLEN ($5 \le MAXLEN \le 80$). Далее следуют данные пользователей. Для каждого пользователя указано его имя, затем, возможно, второе, третье и так далее имя (или же отчество) и затем — фамилия, разделённые единичными пробелами. Имя или фамилия может состоять из строчных и прописных латинских букв, дефисов (‘-’) и апострофов. Гарантируется, что фамилия содержит не менее двух букв, а каждое из имён — не менее одной. Общая длина данных по каждому пользователю не превосходит 80 байт. Входной файл завершается тестовым примером с $U = MAXLEN = 0$, обрабатывать который не требуется.

Формат выходных данных

Для каждого тестового примера выведите в соответствии с форматом, указанным в примере к задаче, список сгенерированных системой логинов.

Пример

standard input	standard output
2 6 John Ax Christos H Papadopulos 11 8 Jean-Louis d'Arnoux Jean-Louis A d'Arnoux Jean-Louis B d'Arnoux Jean-Louis C d'Arnoux Jean-Louis D d'Arnoux Jean-Louis D d'Arnoux Jean-Louis F d'Arnoux Jean-Louis G d'Arnoux Jean-Louis H d'Arnoux Jean-Louis I d'Arnoux Jean-Louis J d'Arnoux 11 9 Jean-Louis d'Arnoux Jean-Louis A d'Arnoux Jean-Louis B d'Arnoux Jean-Louis C d'Arnoux Jean-Louis D d'Arnoux Jean-Louis D d'Arnoux Jean-Louis F d'Arnoux Jean-Louis G d'Arnoux Jean-Louis H d'Arnoux Jean-Louis I d'Arnoux Jean-Louis J d'Arnoux 0 0	Case 1 jax cpapad Case 2 jdarnoux jdarnou1 jdarnou2 jdarnou3 jdarnou4 jdarnou5 jdarnou6 jdarnou7 jdarnou8 jdarnou9 jdarno10 Case 3 jdarnoux jdarnoux1 jdarnoux2 jdarnoux3 jdarnoux4 jdarnoux5 jdarnoux6 jdarnoux7 jdarnoux8 jdarnoux9 jdarnou10

Задача M. King (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

Тридевятое королевство представляет собой прямоугольник, разбитый на единичные квадраты. В одном из этих квадратов расположен замок. Требуется найти минимальное время, за которое король, также стоящий в некотором квадрате, сможет добраться до замка.

Король — как ему и положено — за одну секунду может перейти в любой из восьми соседних квадратов.

Формат входных данных

Входной файл содержит не более $15 \cdot 10^3$ тестовых примеров. Каждый тестовый пример задан в одной строке и состоит из шести целых чисел X, Y, K_X , K_Y , C_X и C_Y. X и Y задают размеры королевства в единичных квадратах ($1 \le R, C \le 100$).

K_X и K_Y — координаты квадрата, на котором находится король, а C_X и C_Y — координаты замка ($1 \le K_X, C_X \le X$; $1 \le K_Y, C_Y \le Y$).

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите одно целое число — наименьшее время, за которое король сможет добраться до замка.

Пример

standard input	standard output
10 10 10 10 1 1	9
2 2 1 1 1 2	1
5 5 3 3 5 5	2
6 6 1 1 1 1	0

Задача N. Nonogram (Division 2 Only!)

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 mebibytes

В японском кроссворде (иначе называемом «нонограмма») используется квадратное поле $n \times n$ клеток. Изображения зашифрованы числами, расположенными слева от строк, а также сверху над столбцами. Числа показывают, сколько групп чёрных клеток находятся в соответствующих строке или столбце и сколько слитных клеток содержит каждая из этих групп (например, набор чисел 4, 8 и 3 означает, что в этом ряду есть три группы: первая — из четырёх, вторая — из восьми и третья — из трёх чёрных клеток). Группы должны быть разделены, как минимум, одной пустой клеткой.

Ваша задача — по заданному изображению построить соответствующий ему японский кроссворд.

Формат входных данных

Входной файл состоит из нескольких тестовых примеров. Каждый тестовый пример начинается с целого числа n ($2 \le n \le 100$) — размера игрового поля. Далее в n строках, каждая из которых содержит ровно n символов, задано поле. Пустая клетка обозначена точкой (‘`.`’), а заполненная — заглавной буквой X (‘`X`’). В конце входного файла идёт тестовый пример с $n = 0$, обрабатывать который не требуется.

Гарантируется, что количество строк во входном файле не превышает 1500.

Формат выходных данных

Для каждого тестового примера выведите $2n$ строк. Первые n строк содержат «строки» японского кроссворда, перечисленные сверху вниз, вторые — «столбцы», перечисленные слева направо. Если в какой-то строке или столбце ни одна клетка не заполнена, выведите 0.

Пример

standard input	standard output
3 XXX .XX .X. 3 X.X ..X X.. 5 ..X.. .XXX. X.X.X ..X.. ..X.. 0	3 2 1 1 3 2 1 1 1 1 1 1 0 2 1 3 1 1 1 1 1 1 1 5 1 1