MIPT and Co Contest
Petrozavodsk, , 2011

List of Problems

Problem	Page	Time limit	Memory limit	Name

Your solution must read the input data from the standard input and write the results to the standard output. Output to the standard error stream is prohibited.

Unless explicitly stated in the problem statements, all input elements may be separated by an arbitrary number of whitespace characters. All input data are correct and satisfy the specification given in the problem statement.

The output of your program must exactly satisfy the output specification in the problem statement.

Problem A. Automaton

Time limit: 2 seconds
Memory limit: 128 MiB

Вам дан недетерминированный конечный автомат без ε-переходов над алфавитом из одного символа. Будем называть два состояния в этом автомате частично эквивалентными, если существует слово алфавита, обработка которого автоматом могла быть закончена в любом из этих состояний или если эти состояния эквиваленты некоторому третьему состоянию. Будем называть два слова почти эквивалентными, если их обработка могла быть закончена в почти эквивалентных состояниях.

Отметим, что слова в однобуквенном алфавите задаются своей длиной. Кроме того, в автомате без ε-переходов над алфавитом из одного символа все переходы задаются только начальным и конечным состоянием автомата (т. к. понятно по какому символу переходим).

Input

В первой строке даны через пробел два натуральных числа – количество состояний автомата n и количество переходов m ($0 < n, m < 200000$). Далее идут m строчек с описанием переходов, в каждой два числа a_i, b_i – начало и конец перехода ($1 \le a_i, b_i \le n$). Начальное состояние автомата всегда идет под номером 1.

В следующей строке число T – количество запросов, далее T строк, в каждой два числа s_i, l_i ($0 \le s_i, l_i \le 10^9$) – длины строк про которые надо ответить, являются ли они почти эквивалентными.

Output

Выведите T строк, в каждой Yes, если строки в запросе почти эквивалентны и No, если нет.

Example

stdin	stdout
1 1 1 1 1 1 2	Yes

Problem B. Bijection

Time limit: 2 seconds
Memory limit: 128 MiB

Пусть дано отображение $f : \mathbb{N} \to \mathbb{N}$. Будем говорить, что число b достижимо из a, если в последовательности $f(a), f(f(a)), f(f(f(a))), \ldots$ хотя бы раз встретилось число b. Сколько существует взаимооднозначных отображений из $\{1, 2, \ldots, n\}$ в $\{1, 2, \ldots, n\}$ (таких, что у каждого элемента ровно один прообраз), таких, что числа $2, 3, \ldots, k$ достижимы из 1?

Input

В одной строке через пробел даны числа n, k, p. $0 < k \leq n < 2^{10000}$, $p < 10^6$ – простое.

Output

Вынесите из ответа максимальную степени p и выведите то, что останется по модулю p.

Example

stdin	stdout
2 1 2	0

Problem C. Cut string

Time limit: 2 seconds
Memory limit: 128 MiB

На циклической ленте записаны маленькие латинские буквы. Для ролевой игры требуется вырезать хотя бы k одинаковых слов. При этом чем длиннее это слово будет, тем интереснее будет игра. Найдите длину наибольшего слова, которое можно вырезать хотя k раз.

Input

Во входном файле задано множество тестовых данных. В каждой строке описан ровно один тест: число k и не пустая строка — буквы записанные на ленте. Ввод заканчивается концом файла. Суммарное количество букв на всех лентах в одном тесте не превосходит 10^5. Все числа во входном файле целые положительные, не превосходящие 10^5.

Output

Для каждых входных данных выведите ответ на поставленную задачу — длину наибольшей строки.

Example

stdin	stdout
4 aaaa 1 aaaa	1 4
2 cabcxab 100 abcaba	3 0

Problem D. Polyhedra

Time limit: 2 seconds
Memory limit: 128 MiB

Мальчик Вася всерьез решил заняться созданием трехмерных моделей на компьютере. Но для начала он решил развить свое трехмерное воображение и научиться делать собственными руками проволочные каркасы. В качестве своей первой модели Вася выбрал выпуклый многогранник. Он попросил своего друга Петю указать в пространстве несколько точек, и хочет сделать модель многогранника, который является выпуклой оболочкой всех выбранных точек. Напомним некоторые определения:

Выпуклое множество – это множество точек в трехмерном пространстве, которое вместе с любыми двумя своими точками содержит целиком и отрезок, соединяющий эти две точки.

Выпуклая оболочка некоторого набора точек – это пересечение всех выпуклых множеств, содержащих данный набор.

Ваша задача - определить, сколько прямолинейных кусков проволоки понадобится Васе, чтобы собрать модель многогранника.

Input

Первая строка входного файла содержит целое число $N(4 \le N \le 300)$ - это количество точек, которые выбрал Петя.

Каждая из следующих N строк содержит по три целых числа X, Y, Z, разделенные пробелами - координаты одной из выбранных Петей точек. Все числа по модулю не превосходят 1000.

Гарантируется, что выпуклая оболочка всех указанных во входе точек действительно является невырожденным (имеющим ненулевой объем) многогранником. Одна и та же точка может встречаться во входе произвольное число раз.

Output

Одно целое число - количество ребер у выпуклой оболочки всех указанных точек.

Example

stdin	stdout
8 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	12
4 0 0 0 0 0 1 0 1 0 1 0 0	6

Problem E. Egypt Tale

Time limit: 1 seconds
Memory limit: 128 MiB

В последнее время, Вася очень интересуется египетской культурой. В интернете он прочитал, что древних египтян больше всего интересовали две вещи – математика и пирамиды. На том же сайте он узнал, что его любимый ученый Леонардо Пизанский в молодости учился в Египте. «Вот много же совпадений на свете!», - удивился Вася.

Вася еще немало времени провел после этого в интернете, читая разные математические статьи. Поэтому он уснул на клавиатуре, а это всегда приводит к неприятным последствиям. В данном случае ему приснился сон, в котором он сидел в белом чурбане, на берегу моря, из которого кровожадно выглядывали акулы, перед ним на песке лежал ноут и надо было решить странную задачу...

«Приснится же такое», – подумал Вася. – «хотя, а вдруг ...» и сел обратно за клавиатуру.

Input

В первой строке дано количество тестов T ($0 < T \leq 500$). Каждый тест имеет следующий формат:

Первая строка начинается с высоты пирамиды n ($0 < n \leq 10$). Далее, в той же строке идут n чисел a_i ($0 < a_i \leq 10^9$), образующих пирамиду

$$N = a_1^{a_2^{a_3^{\cdots}}}$$

Во второй строке находится число m ($0 < m < 2^{32}$)

Output

Выведите для каждого теста F_N по модулю m, где F_n – числа Фибоначчи, $F_0 = 0, F_1 = 1, F_{n+2} = F_n + F_{n+1}$. В конце вывода каждого теста делайте перевод строки.

Example

stdin	stdout
2 2 2 2 13 2 2 3 13	3 8

Problem F. Probki

Time limit: 2 seconds
Memory limit: 128 MiB

Крупнейшая в тридевятом царстве компания, производящая GPS-навигаторы, предложила вам написать программу, которая определяла бы оптимальный маршрут движения автотранспорта в городских условиях. После недели мучительных разбирательств с интерфейсом, форматом статистических данных и уже написанным сотрудниками компании кодом, вам удалось формализовать, что от вас требуют.

Город состоит из перекрестков, которые соединены между собой односторонними дорогами. Для каждой дороги задан показатель времени движения по дороге, округленный до минут. В зависимости от времени суток этот показатель может быть различным (может меняться плотность потока машин, образовываться пробки и т. д.). В результате анализа статистических данных, собранных сотрудниками компании, была построена следующая модель – для каждой дороги есть несколько моментов времени в течение суток, в которые может происходить "перелом характера движения". Статистические данные были собраны с точностью до получаса, поэтому во всех данных моменты времени, когда произошел перелом, также округлены до получаса. Между переломными моментами время движения по дороге изменяется линейно раз в пять минут (т. е. если начать двигаться по дороге в $18:00, 18:01, 18:02, 18:03$ или $18:04$, то время движения будет t, если выехать в $18:05 - 18:09$, то время движения будет равно $t + a$, если выехать в $18:10 - 18:14$, то время движения будет $t + 2a$).

Пока, к счастью, вам не нужно писать программу для навигации в столице, для начала вам дали данные нескольких небольших городов для тренировки. Ваша задача – уметь быстро отвечать на запросы: как быстрее всего добраться от перекрестка, где находится дом клиента (он всегда выезжает из дома в 8.00 утра), до некоторого перекрестка в городе и как быстрее добраться от перекрестка, где находится работа клиента (он всегда уезжает с работы в 18.00), до некоторого перекрестка в городе. Время проезда по перекрестку уже учтено в данных, поэтому считайте его равным нулю.

Input

В первой строке даны два числа – количество перекрестков ($0 < n < 100000$) и количество дорог в городе ($0 \le m < 100000$). Далее в m строках идут описания дорог: два числа a_i, b_i – номера перекрестков, откуда и куда идет дорога ($1 \le a_i, b_i \le n$), затем время, которое потребуется для перемещения по дороге в 00:00, затем число q_i ($0 \le q_i \le 10$) – количество переломных моментов времени, затем идет q_i описаний каждого из переломных моментов – время в формате чч:мм (всегда округленное до получаса) и целое число – коэффициент изменения времени движения (на сколько минут дольше или короче будет становиться проезд по дороге через каждые 5 минут)

В следующей строке даны два натуральных числа не превосходящих n – индексы перекрестков, на которых находятся дом и работа клиента.

В следующей строке дано количество запросов T ($0 \le T \le 10000$). Далее в T строках даны описания запросов: два числа x_i (x_i – ноль, если надо доехать из дома или один, надо доехать от работы), y_i – номер перекрестка, до которого надо добраться.

В каждой строке числа и строки разделены пробелами.

Все данные о переломных моментах корректны, т. е. время движения по каждой дороге однозначно определяется в любое время суток, причем длина дороги всегда натуральное число, не превосходящее 1000. Все данные о временах движения по дорогам удовлетворяют правилу "ожи-

дания": если $t_1 < t_2$ – некоторые моменты времени в абсолютной шкале, а $T(t_1)$ и $T(t_2)$ – времена движения по дороге, если выехать в t_1 или t_2, соответственно. Тогда $t_1 + T(t_1) \leq t_2 + T(t_2)$.

Output

Выведите T строк, в каждой ближайший момент времени, в который можно добраться до нужного перекрестка во формате чч:мм или «Impossible» (без кавычек), если добраться туда нельзя.

Example

stdin	stdout
2 2 1 2 10 0 2 1 13 0 1 2 2 0 1 1 1	0 13

Problem G. Graph Game

Time limit: 2 seconds
Memory limit: 128 MiB

Алиса и Боб играют в игру со следующими правилами. В начале есть граф ребер на N вершинах. Каждым ходом игроки соединяют ребром две вершины, которые до сих не были соединены (т.е. кратных ребер не бывает). Игрок, после чьего хода граф становится связным проигрывает. Первым ходит Боб.

Input

В первой строке количество тестов T ($T < 100$) Следующие T строк содержат одно число $N_i \geq 2$ – количество вершин в графе. Сумма всех N не превосходит 200000.

Output

Выведите T строк, в каждой должно быть написано имя победителя в соответсвующей игре.

Example

stdin	stdout
2 2 3	Alice Bob
1 10	Bob

Problem H. Politicans

Time limit: 2 seconds
Memory limit: 128 MiB

Депутаты Государственной Думы Российской Федерации решили устроить банкет по поводу ухода Думы на каникулы. Известно, что на банкете будут присутствовать N депутатов с левыми политическими взглядами и M депутатов - с правыми. В банкетном зале имеется один круглый стол как раз на $N + M$ мест. Вам было поручено рассадить депутатов за этим круглым столом. При этом Вы хотите, чтобы беседа за обедом проходила мирно и без конфликтов. Для того, чтобы этого добиться, Вы должны быть уверены, что нигде не образуется переконцентрация людей с теми или иными политическими убеждениями. Поэтому нельзя сажать подряд больше, чем A левых или B правых политиков.

Input

Единственная строка входного файла содержит 4 числа N, M, A, B, разделенные пробелами. Все числа лежат в промежутке от 1 до 1000 включительно.

Output

Количество способов рассадить депутатов за круглым столом на $N + M$ мест. Поскольку ответ может получиться достаточно большим, выведите только остаток от его деления на 10^9.

Example

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

Problem I. Robots

Time limit: 2 seconds
Memory limit: 128 MiB

У одной известной компании, занимающийся торговлей спортивных товаров, есть склад размером $m \times n$. Для склада предусмотрены автоматические роботы, каждый размером ровно с одну клетку, которые умеют перемещаться по складу. Робот может ходить из клетки в одну из четырех соседних. При этом среди клеток склада есть те, по которым робот ходить не может из-за конструктивных особенностей склада.

Руководство компании решило использовать несколько роботов так, чтобы под их наблюдением находились все доступные клетки склада. Для простоты было решено, что каждый робот ходит по определенному замкнутому маршруту (циклу длины не менее 3 клеток). При этом в этом замкнутом маршруте никакая клетка не может встретиться дважды.

Однако, в процессе эксплуатации оказалось, что роботы часто сталкивались. Теперь руководство предприятия просит вас по схеме склада составить расписание для роботов так, чтобы маршруты не пересекались, но все свободные клетки склада были покрыты маршрутом одного из роботов.

Input

Первая строка содержит два целых числа m и n ($1 \le m, n \le 50$).

Следующие m строк содержат по n чисел, каждое либо 0, либо 1 (0 – означает свободную клетку, 1 – занятую).

Output

Выведите в выходной файл либо Yes либо No, в зависимости от того можно ли составить маршруты роботов или нет.

Example

stdin	stdout
3 3 1 0 0 0 0 0 0 0 0	Yes
3 3 1 0 0 0 0 0 0 0 1	No

Problem J. Trees

Time limit: 3 seconds
Memory limit: 128 MiB

Напомним, что деревом называется связный неориентированный граф, в котором отсутствуют циклы.

Плоским деревом мы назовем дерево, изображенное на плоскости. При этом вершины изображаются различными точками, ребра – отрезками прямых линий, соединяющими соответствующие вершины, и эти отрезки могут пересекаться только по своим концам.

Два различных плоских дерева будем называть изоморфными, если одно дерево можно перевести в другое непрерывной деформацией. В ходе этой деформации дерево должно постоянно оставаться плоским деревом, то есть вершины должны оставаться различными точками, а ребра должны оставаться отрезками прямых и пересекаться только по своим концам.

Напишите программу, которая для двух заданных плоских деревьев найдет в них пару изоморфных плоских поддеревьев (по одному поддереву в каждом из исходных деревьев) с максимальным количеством вершин.

Input

Во входном файле содержатся описания двух плоских деревьев. Каждое плоское дерево описывается следующим образом: сначала идет строка с единственным целым числом N – количеством вершин в дереве. Следующие N строк описывают ребра дерева. i-я строка описывает ребра, имеющие своим концом i-ю вершину, в ней записаны числа, разделенные пробелами, $v_1, v_2, \ldots, v_k$ – номера вершин, соединенных ребрами с i-й вершиной, перечисленные в порядке движения по часовой стрелке ($0 \le k \le N - 1$). Каждое из описаний оканчивается нулем.

Количество вершин в каждом графе лежит в пределах от 1 до 200 включительно. Вершины в каждом графе нумеруются последовательными натуральными числами, начиная с 1.

Output

Единственная строка с одним целым числом M - количеством вершин в максимальных изоморфных плоских поддеревьях исходных плоских деревьев.

Example

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

Problem K. Triangle and circle

Time limit: 3 seconds
Memory limit: 128 MiB

Вам кажется, что вы любите решать геометрические задачи? Тогда попробуйте найти пересечение круга и треугольника на плоскости.

Input

В первой строке заданы x_0, y_0, r_0 — центр и радиус окружности. Во второй строке заданы $x_1, y_1, x_2, y_2, x_3, y_3$ – координаты вершин треугольника. Все числа во входном файле – целые, положительные и не превосходят 100.

Output

Выведите единственное число — площадь пресечения круга и треугольника с абсолютной или относительной погрешностью точностью 10^{-6}

Example

stdin	stdout
1 10 10 1 1 1 2 2 1	0.5000000000
10 10 10 1 1 1 2 2 1	0.0000000000