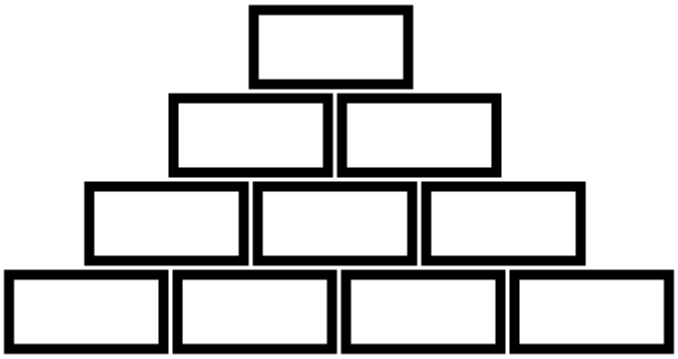
Задача A. Кирпичи

Имя входного файла:	`bricks.in`
Имя выходного файла:	`bricks.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Ежегодная распродажа кирпичей, проходящая 32 марта, в самом разгаре. Сотни людей уже стоят в очередях, ожидая открытия кирпичных магазинов по всему миру, а самая длинная очередь, как обычно, выстроилась к супермаркету кирпичей BrickStore. Известность и популярность этот магазин приобрел благодаря своему товару — уникальным кирпичам идеальной формы прямоугольного параллелепипеда с центром тяжести, точно совпадающим с геометрическим центром кирпича.

По традиции, кирпичи к распродаже выстраиваются в огромную симметричную стену, нижний уровень которой состоит из n кирпичей, лежащих на полу. Кирпичи расположены так, как показано на рисунке ниже:



Каждый кирпич задаётся двумя координатами — номером ряда (ряды нумеруются сверху вниз) и позицией кирпича в этом ряду слева направо. Каждый из k покупателей в очереди точно знает, за каким конкретным кирпичом он пришел и готов купить только его. При этом конфликты интересов в очереди исключены — у покупателей достаточно времени пообщаться, а значит, не может быть двух клиентов, жаждущих одного кирпича. При покупке кирпич вытаскивается из стены настолько осторожно, что другие кирпичи никак не меняют своего положения.

Инженер по технике безопасности супермаркета выяснил, что если вдруг в определенный момент времени центр тяжести одного из кирпичей не будет опираться на границу какого-либо другого кирпича или на пол, то первый упадет. Эта ситуация недопустима, поэтому ответственный сотрудник обошел всю очередь, и каждый покупатель сообщил ему два числа (x, y) — координаты кирпича, который он приобретет.

Все что остается сделать — определить номер первого покупателя в очереди, покупка которого грозит падением хотя бы одного кирпича, чтобы успеть принять меры.

Формат входного файла

В первой строке — два целых числа $1 \le n \le 10^9, 1 \le k \le 2 \cdot 10^5$. В каждой из последующих k строк по паре целых чисел (x_i, y_i) — координаты кирпича, который собирается купить i-ый покупатель.

Формат выходного файла

Единственное число — номер искомого покупателя в очереди. Если распродажа пройдет без падений, вывести `-1`.

Примеры

bricks.in	bricks.out
10 9 1 1 2 1 2 2 4 1 4 3 5 3 5 4 4 2 5 5	8

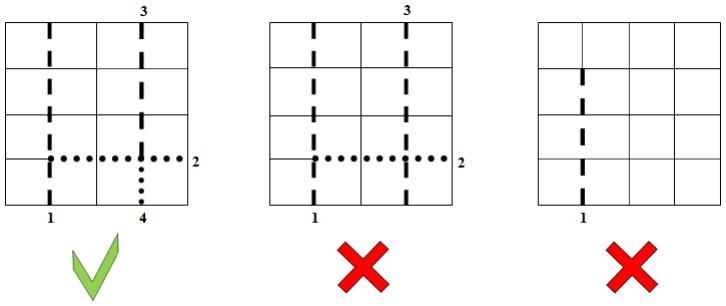
Задача B. Разрезы

Имя входного файла:	`cuts.in`
Имя выходного файла:	`cuts.out`
Ограничение по времени:	3 seconds
Ограничение по памяти:	256 megabytes

В поездах пассажирам предлагается множество способов убить время. На смену устаревшим кроссвордам приходят новые изощренные головоломки. Одна из таких создана специально для пассажиров поезда Анкоридж-Казань.

Головоломка представляет собой клетчатый лист бумаги размера $n \times m$. В каждой клетке записано единственное число от 1 до $n \cdot m$. При этом каждое число используется ровно один раз.

Задача — разрезать этот лист на квадратики 1×1. Каждый разрез обязан начинаться с одной стороны листа, заканчиваться другой стороной и быть параллельным одной из координатных осей. Примеры разрезов приведены ниже.



Определим время освобождения квадратика как количество разрезов, проведенных до того, как выбранный квадратик окажется освобожден от своих соседей и будут освобождены все квадратики с меньшим номером.

Нас интересуют такие способы разрезания, при которых квадратик, содержащий 1, оказывается освобожден за минимальное количество разрезов. Из всех таких способов выбираются способы самого раннего освобождения квадратика, содержащего 2, и так далее. Самым низким приоритетом обладает квадратик, содержащий $n \cdot m$.

Необходимо найти $n \cdot m$ чисел — времена освобождения квадратика с единицей, с двойкой, ..., с $n \cdot m$ при оптимальном порядке выполнения разрезов.

Формат входного файла

В первой строке записаны два целых числа n и m ($1 \le n, m \le 30$). Далее следуют n строк по m чисел в каждой, описывающие головоломку.

Формат выходного файла

Вывести $n \cdot m$ искомых чисел.

Примеры

cuts.in	cuts.out
2 3 1 6 4 3 5 2	2 3 4 5 5 5

Задача C. Сосульки (Div-1 only!)

Имя входного файла:	`ice.in`
Имя выходного файла:	`ice.out`
Ограничение по времени:	3 seconds
Ограничение по памяти:	256 megabytes

С крыши, расположенной очень высоко, свисают n квадратных сосулек со стороной 1. Центры сосулек лежат на одной прямой, параллельной оси абсцисс, при этом i-ая сосулька расположена на отрезке $[i-1, i]$ по оси абсцисс.

В воздухе находятся m воздушных шариков. Центры шариков лежат в одной плоскости со всеми сосульками, поэтому каждый шарик обладает тремя параметрами: положением центра по оси абсцисс и оси ординат, радиусом. Все шарики двигаются вверх вдоль оси ординат с одинаковой скоростью, проходя за одну единицу времени одну единицу измерения длины. Любая пара шариков имеет не более одной общей точки, при этом шарики не могут находиться внутри друг друга.

Когда хотя бы один из шариков коснется сосульки, наступит Конец Света. Возьмем этот момент времени за точку отсчета.

Девочка Валя приобрела лазер и теперь полна решимости расплавить некоторые сосульки, чтобы отсрочить Конец Света, ведь расплавленная сосулька не несет никакой угрозы шарикам.

Валя не знает, на сколько выстрелов хватит энергии лазера, но точно уверена, что спасти все шарики не удастся. Поэтому можно предполагать, что все неотрицательные количества выстрелов, не позволяющие предотвратить Конец Света, равновероятны. Валя готова действовать оптимальным образом, то есть выбрать такую последовательность уничтожения сосулек, которая позволит отсрочить Конец Света на максимально возможное время.

Необходимо определить ожидаемое время отсрочки Конца Света.

Меткая Валя никогда не промахивается, а лазер всегда стреляет. За один выстрел лазер расплавляет ровно одну сосульку. Гарантируется, что у Вали достаточно времени, чтобы сделать все выстрелы до того, как шарики приблизятся к сосулькам на опасное расстояние.

Формат входного файла

В первой строке указано количество сосулек n и количество шариков m, $1 \le n \le 10^7$, $1 \le m \le 40\,000$.

В последующих m строках содержатся тройки целых чисел x, y, r, где x, y — координаты центра шарика, а r — его радиус, $-10^9 \le x, y \le 10^9$, $1 \le r \le 10^9$.

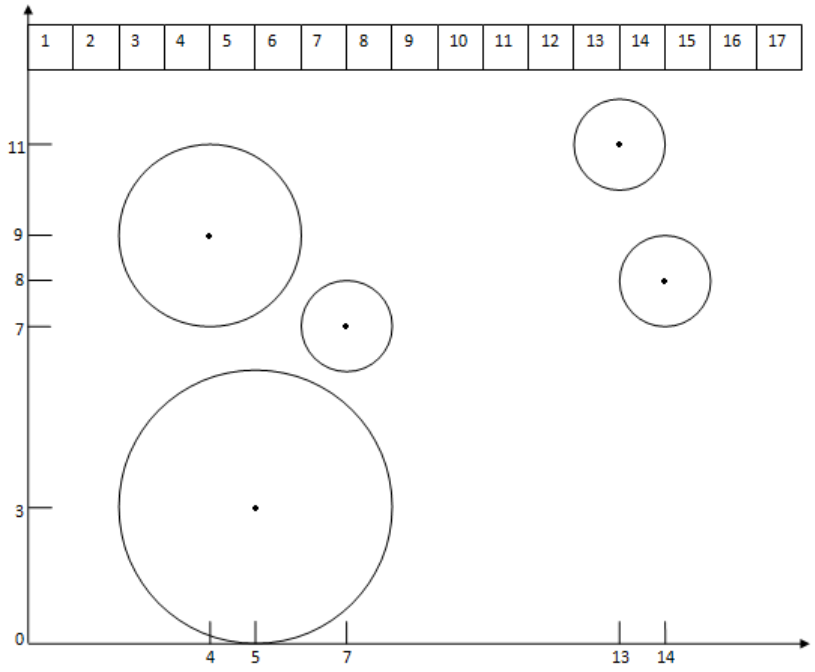
Формат выходного файла

Необходимо вывести единственное число — ожидаемое время, на которое Валя сможет отсрочить Конец Света. Абсолютная или относительная погрешность ответа не должна превышать 10^{-6}.

Примеры

ice.in	ice.out
17 5 5 3 3 4 9 2 7 7 1 13 11 1 14 8 1	1.8239927417758746723

Замечание



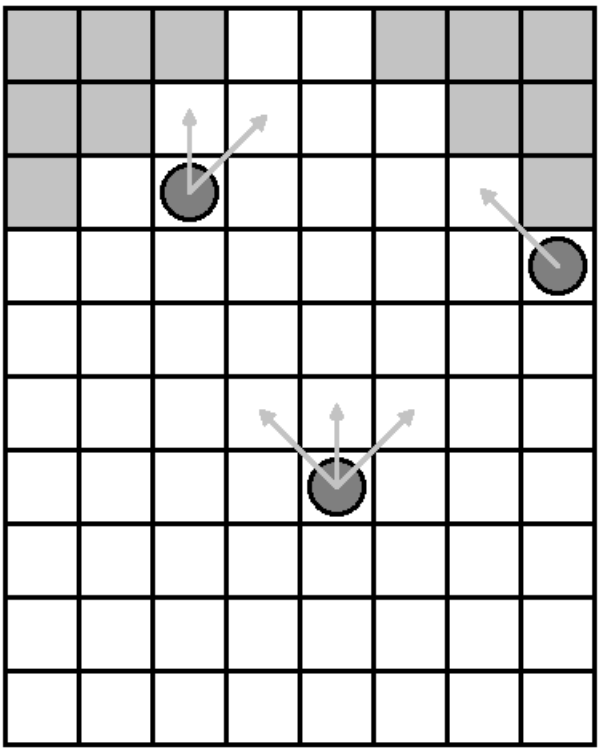
Задача D. Миллениум

Имя входного файла:	`millenium.in`
Имя выходного файла:	`millenium.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Пробки на дорогах возникают по разным причинам. Часто они образуются при въезде на мост по той простой причине, что число полос для проезда транспортных средств по мосту меньше, чем число полос, по которым эти транспортные средства подъезжают. Для иллюстрации этого эффекта в Казани построили мост, который назвали «Миллениум», что в переводе с древне-казанского означает «бесконечная пробка».

Всего в Казани N автомобилей, и так уж получилось, что все они оказались в пробке перед въездом на мост «Миллениум».

Участок дороги при въезде на мост можно представить в виде матрицы высотой H строк и шириной $2 \cdot A + B$. На рисунке пример такой дороги с $H = 10$, $A = 3$, $B = 2$.



Каждая из машин в этой матрице изначально занимает одну из клеток. Заштрихованные клетки заняты отбойниками перед сужением дороги. Таким образом, ширина дороги в первой строке равна B столбцов, во второй $B+2$ и так далее. Начиная с $A+1$-ой строки, ширина дороги равна $2 \cdot A + B$.

Машины движутся снизу вверх и должны въехать на мост через узкую часть дороги шириной B столбцов. За одну единицу времени каждая машина может продвинуться на одну клетку в одном из трех направлений: влево вверх, вверх или вправо вверх, при условии, что клетка свободна. В случае необходимости машина может приостановить движение на одну или несколько единиц времени. Машины двигаются ритмично и синхронно. Это означает, что если какая-то машина освободила свое место, то одна из стоящих следом за ней машин может занять освободившееся место в ту же единицу времени.

Водители требуют информации о том, как быстро все машины смогут покинуть участок сужения дороги, проехав на мост, при оптимальном согласованном действии всех водителей.

Формат входного файла

В первой строке три целых числа $1 \le N \le 10^5$, $1 \le A \le 10^9$ и $1 \le B \le 10^9$.

Следующие N строк содержат по два целых числа x и y — номер строки и номер столбца i-го автомобиля, $1 \le x \le 2 \cdot A + B$ и $1 \le y \le 10^9$. Гарантируется, что ни одна машина не находится в области, занятой отбойниками.

Формат выходного файла

Вывести время, за которое все автомобили смогут проехать на мост.

Примеры

millenium.in	millenium.out
5 2 2 1 3 1 4 2 2 2 3 2 4	3

Задача E. Фифекты гечи

Имя входного файла:	`changes.in`
Имя выходного файла:	`changes.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Единственная в мире группа этнографов-логопедов обнаружила затерянный остров. К великому счастью первооткрывателей, остров был заселен людьми, каждый из которых имел ровно один дефект речи. Всего ученым стало известно m различных дефектов речи, встречавшихся на острове. Кроме того, было обнаружено необычное свойство острова: при встрече $(m-1)$ островитян с разными дефектами речи они все одновременно избавляются от своего дефекта, но приобретают дефект, не представленный на встрече. К примеру, при $m = 3$ шепелявый и картавый, встретившись, начинают заикаться. Во всех других случаях наблюдаемые не меняют своего дефекта.

В настоящее время специалисты выясняют точную численность островитян, пытаясь решить жизненно важный вопрос: может ли случиться, что в некоторый момент на острове окажется ровно l_1 людей с первым дефектом речи, l_2 — со вторым, и так далее, наконец, l_m людей, имеющих m-ый дефект речи, и, если может, то при каких обстоятельствах.

Формат входного файла

Первая строка содержит одно целое m $(3 \le m \le 10\,000)$ — количество дефектов речи. Вторая строка содержит m целых чисел $k_1, k_2, ..., k_m$, где k_i — количество людей с i-ым дефектом, проживающих на острове $(0 \le k_i \le 10\,000)$. В следующей строке записаны m целых чисел $l_1, l_2, ..., l_m$ — количества людей для каждого дефекта, которые требуется получить с помощью подходящей последовательности встреч $(0 \le l_i \le 10\,000)$. Гарантируется, что последовательности различны.

Формат выходного файла

В первой строке запишите одно число — наименьшее количество встреч, после которых достигается требуемое распространение дефектов. Во второй строке запишите последовательность разделенных пробелом чисел, каждое из которых задает номер дефекта, не представленного в очередной встрече. Если возможных решений несколько, выведите любое из них. Если решений не существует, выведите `-1`.

Примеры

changes.in	changes.out
3 3 0 0 0 0 3	-1
3 1 2 3 2 3 1	2 1 2

Замечание

Пояснение ко второму примеру. Наименьшее число встреч равно двум. После первой встречи, в которой не участвуют люди с первым дефектом, количества людей со вторым и третьим дефектом уменьшатся на единицу, а число людей с первым дефектом увеличится на два, что будет описываться тройкой $(3, 1, 2)$. После второй встречи, в которой не было людей со вторым дефектом, получим требуемую тройку $(2, 3, 1)$.

Задача F. Апельсины (Div-1 only!)

Имя входного файла:	`oranges.in`
Имя выходного файла:	`oranges.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Ученым, наконец, удалось вывести сорт апельсинов, в которых нет косточек. Однако радовались они этому недолго. Очень скоро стало ясно, что если в апельсинах нет косточек, то неоткуда взяться новым деревьям с плодами того же сорта.

Вскоре ученые провели еще несколько экспериментов и немного модифицировали сорт — теперь на каждом дереве будут вырастать несколько апельсинов без косточек и ровно один особенный красный апельсин, который созреет лишь однажды — вместе с первым урожаем, и в нем будут скрыты ровно две косточки.

Фруктовая компания «O'Range» купила у ученых одну косточку нового сорта и начала засаживать плантации апельсиновыми деревьями. Через некоторое время работники компании заметили несколько интересных особенностей. Во-первых, на любом дереве каждый сезон созревает одно и то же количество апельсинов без косточек. В частности, первое посаженное на плантации дерево в каждый урожай дает всего один апельсин без косточек. Во-вторых, если с дерева, на котором растет x апельсинов без косточек, сорвать красный апельсин и из его косточек вырастить два других дерева, то на одном из них будет $(p \cdot x)$ апельсинов без косточек, а на другом — $(p \cdot x - 1)$. И в-третьих, красные апельсины с деревьев, на которых растет больше чем $(2 \cdot n)$ апельсинов, содержат нежизнеспособные косточки, а значит, из них не вырастет дерево.

Плантации разрослись, и сейчас на них растут все деревья, которые можно было вырастить из одной косточки.

Компания отгружает апельсины бочками по n штук. Причем фирменной особенностью торговой марки является наличие в каждой упаковке урожая с двух деревьев. Теперь менеджменту компании предстоит выяснить, сколькими способами можно выбрать пару апельсиновых деревьев так, чтобы за сезон на них суммарно созревало ровно n апельсинов без косточек.

Формат входного файла

В единственной строке записаны два целых числа p и n ($2 < p < 10^9$, $0 < n < 10^{2000}$).

Формат выходного файла

Выведите одно число — ответ на задачу по модулю $10^9 + 7$.

Примеры

oranges.in	oranges.out
3 7	2
5 20	1

Задача G. Догонялки на графе

Имя входного файла:	`runaway.in`
Имя выходного файла:	`runaway.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Первое в мире реалити-шоу для программистов «Догонялки» снимается в графе с n вершинами и m рёбрами одинаковой длины. В шоу участвуют три игрока: в вершине $1 \le A \le n$ находится Алиса, в вершине $1 \le B \le n$ находится Боб, в вершине $1 \le C \le n$ — Цезарь. Игроки одновременно начинают движение из исходных вершин и перемещаются по ребрам с постоянной одинаковой скоростью.

Цель Алисы — догнать Боба и не быть пойманной Цезарем, цель Боба — догнать Цезаря и не быть пойманным Алисой, цель Цезаря — догнать Алису и не быть пойманным Бобом. Поимка происходит тогда, когда два или более игроков оказываются в одной точке, причем не обязательно в вершине. В этот момент игра заканчивается. Игрок, достигший своей цели, объявляется победителем.

Игроки не обязаны двигаться и могут останавливаться и продолжать движение в произвольные моменты времени.

Все игроки эгоистичны, асоциальны, фанатичны и рассуждают независимо друг от друга абсолютно одинаково, пытаясь воспользоваться наилучшей тактикой. Эгоистичность означает, что каждого игрока устраивает только победа (формально, ситуации быть пойманным и не поймать оппонента считаются одинаковыми поражениями). Асоциальность говорит о том, что никакая пара игроков не способна объединиться против третьего. Фанатизм означает, что даже в случае очевидно неотвратимого поражения игрок будет сопротивляться так долго, как это только возможно.

Таким образом, самый приоритетный результат — это победа, следующий за ним — ничья, и самый нежелательный — поражение по любой из двух причин.

Продюсерам шоу срочно нужно выяснить, кто выиграет, если игроки будут играть оптимальным образом. Если игра будет длиться бесконечно долго, или все игроки окажутся в одной точке, результатом будет ничья.

Формат входного файла

В первой строке содержится единственное число T — количество тестов, не превышающее 10 000. Каждый из тестов имеет следующий вид: в первой строке теста записана пара целых чисел n и m ($3 \le n \le 10^5$, $0 \le m \le 10^5$). Далее следуют m строк по паре целых чисел x, y ($1 \le x, y \le n$), описывающих ребра графа. В последней строке теста записано три различных целых числа A, B и C — номера вершин Алисы, Боба и Цезаря соответственно.

Гарантируется, что суммарное количество вершин во входном файле не превышает 10^5, и суммарное количество рёбер во входном файле не превышает 10^5.

Формат выходного файла

Выведите T строк. i-ая строка должна содержать имя победителя в i-ом тесте (`ALICE`, если выиграет Алиса, `BOB`, если выиграет Боб, `CAESAR`, если выиграет Цезарь) или `DRAW` в случае ничейного результата.

Примеры

runaway.in	runaway.out
3 5 4 1 2 2 3 3 4 4 5 1 3 5 4 3 1 2 1 3 1 4 2 3 4 3 3 1 2 1 3 2 3 1 2 3	BOB DRAW DRAW

Задача N. Тополя

Имя входного файла:	`trees.in`
Имя выходного файла:	`trees.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Мэр города Топольска очень любит тополя и сажает их по всему городу. А еще он любит порядок, поэтому все улицы города подчиняются одному правилу — между всеми парами соседних деревьев на улице должно быть строго одинаковое расстояние. Это правило соблюдается неукоснительно, поэтому у топольского ГорЗеленХоза уже построены за городом огромные питомники, готовые обеспечить город любым количеством тополиных саженцев.

Проектировщики новой улицы длиной n километров разделили её переулками на отрезки длиной 1 километр. Помня о странностях своего мэра, в центре каждого отрезка между переулками они предусмотрели участок газона, на котором возможна посадка одного тополя. Но не успели городские службы приступить к работе, а на k участках уже выросли тополя!

К сожалению, вырубать выращенные тополя строго запрещено. Теперь коммунальщикам стало интересно, сколько у них есть способов рассадить тополя по оставшимся участкам так, чтобы соблюсти “правило Мэра“.

Формат входного файла

В первой строке два числа: $1 \le n \le 2 \cdot 10^9$, $1 \le k \le 10^5$. Во второй строке k чисел, отсортированных по возрастанию: позиции уже посаженных деревьев.

Формат выходного файла

Вывести единственное число — количество способов “правильной” рассадки тополей.

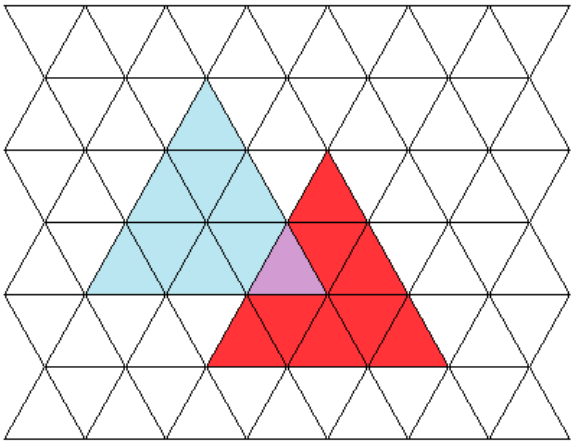
Примеры

trees.in	trees.out
4 2 1 4	2
3 1 2	4

Задача I. Треугугль (Div-1 only!)

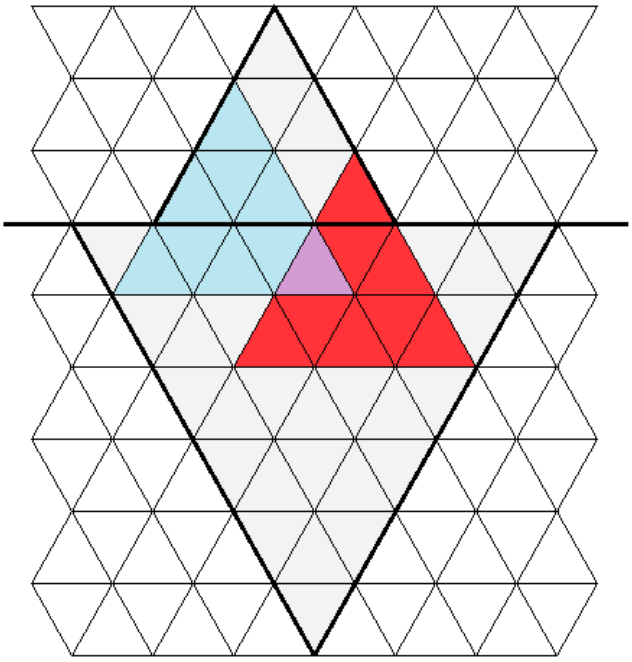
Имя входного файла:	`queries.in`
Имя выходного файла:	`queries.out`
Ограничение по времени:	4 seconds
Ограничение по памяти:	256 megabytes

Поисковая система треугугль, которую потребуется реализовать в этой задаче, обрабатывает поисковые запросы по бесконечному треугольному полю, на котором расположено n треугольников, возможно, имеющие пересечения.



Необходимо обработать k запросов. Каждый запрос — прямая, которая разрезает исходную плоскость на две полуплоскости. Эта прямая называется поисковой. Некоторые треугольники могут оказаться разрезанными поисковой прямой. Нужно определить, как построить два треугольника (возможно вырожденных) таким образом, чтобы каждый из них находился целиком в своей полуплоскости, одна из сторон лежала на поисковой прямой, и все разрезанные части треугольников полностью оказались покрытыми этими двумя треугольниками. При этом важно минимизировать суммарную площадь построенных треугольников. Каждый запрос применяется к исходному состоянию плоскости.

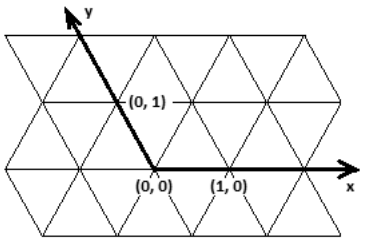
Ниже изображено, как нужно построить треугольники по обе стороны плоскости.



Формат входного файла

В первой строке записано два целых числа $1 \le n \le 40\,000$, $1 \le k \le 10^5$.

В следующих n строках содержится по шесть чисел: координаты вершин треугольников. Координаты определяются так, как показано на рисунке ниже.



Гарантируется, что все треугольники невырожденные, правильные, их стороны параллельны трём осям.

В последних k строках содержится по четыре числа: координаты двух различных точек, через которые проходит поисковая прямая. Гарантируется, что поисковая прямая параллельна одной из трёх осей.

Все координаты целые и не превосходят 10^8 по модулю.

Формат выходного файла

Для каждого из k запросов выведите в отдельной строке одно число — минимальную суммарную площадь построенных треугольников, покрывающих все разрезанные треугольники.

Примеры

queries.in	queries.out
2 4 1 1 1 -2 -2 -2 0 -1 -3 -1 0 2 -2 0 0 0 -1 1 0 1 1 1 2 2 1 1 1 0	45 26 26 0

Задача J. Острова

Имя входного файла:	`islands.in`
Имя выходного файла:	`islands.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Архипелаг «Острова зеленого крыса» представляет собой сетку $n \times m$, в узлах которой находятся небольшие острова. Два острова являются соседними, если евклидово расстояние между ними не больше 1.

В центре каждого острова находится органайзер, содержащий в себе k мостов ($0 \le k \le 9$). Число k может различаться на разных островах и называется мостовым числом острова. Автоматическая система, управляемая пультом, может либо устанавливать любой из мостов, содержащихся в органайзере, на пролет до соседнего острова, либо возвращать их обратно в органайзер. Любой мост может быть возвращен независимо от остальных, причем только в тот органайзер, из которого изначально брался. Мосты могут соединять только соседние острова. Между двумя островами можно перемещаться, только если они соединены мостом. Пара соседних островов может быть соединена более чем одним мостом.

Зеленый крыс (единственный житель архипелага) перемещается между островами, управляя мостами с помощью пульта. Зеленый сейчас находится на острове A и хочет выяснить, сможет ли он добраться до острова B. Ответить на этот вопрос было бы легко, если бы не законодательство островов, согласно которому перемещаться по одному и тому же мосту два раза подряд категорически запрещено.

Формат входного файла

В первой строке записаны два целых числа $1 \le n, m \le 10^5$, $n \cdot m \le 10^5$.

Последующие n строк содержат по m цифр — мостовые числа соответствующих островов.

В последних двух строках заданы пары координат островов A и B соответственно, где первое число указывает строку, а второе — столбец сетки.

Формат выходного файла

Выведите `YES`, если возможно добраться с острова A на остров B, и `NO` — в противном случае.

Примеры

islands.in	islands.out
4 6 011110 100011 110001 011112 1 1 2 2	YES
3 3 201 010 030 2 1 2 3	YES
3 3 200 000 002 1 1 3 3	NO

Замечание

Пояснение ко второму примеру — путь с острова A на остров B выглядит следующим образом:

$$(2,1) \to (1,1) \to (1,2) \to (2,2) \to (3,2) \to (2,2) \to (2,3).$$

Задача K. Подпалиндромы (Div-1 only!)

Имя входного файла:	`palindromes.in`
Имя выходного файла:	`palindromes.out`
Ограничение по времени:	4 seconds
Ограничение по памяти:	256 megabytes

Палиндромом называется символьная строка, одинаково читающаяся как слева направо, так и справа налево. Палиндромологией называется наука, занимающаяся поиском палиндромов везде, где только можно их найти. Сегодня поиск ведется в некоторой строке S, состоящей из строчных латинских букв.

Пусть $S[i, j]$, $i \le j$ обозначает подстроку с i-ого символа по j-ый. Для подстроки $S[a, b]$ необходимо вычислить количество подстрок-палиндромов. Другими словами, нужно найти количество $S[x, y]$, таких что:

- $a \le x \le y \le b$,
- $S[x, y]$ является палиндромом.

Формат входного файла

В первой строке записана исходная строка S, длина которой не превосходит 10^5 символов. В следующей строке — количество запросов m ($1 \le m \le 3 \cdot 10^5$). Далее следуют m строк, в каждой из которых записано по паре целых чисел a и b ($1 \le a \le b \le n$, где n — длина строки S), обозначающие запрос на поиск подстрок-палиндромов в подстроке $S[a, b]$.

Формат выходного файла

На каждый запрос выведите ответ в отдельной строке.

Примеры

palindromes.in	palindromes.out
abaa 4 1 2 1 3 1 4 3 4	2 4 6 3

Задача L. Гонки

Имя входного файла:	`race.in`
Имя выходного файла:	`race.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Далеко в затерянном пространстве на плоской планете проводятся необычные автогонки. Необычны они тем, что автомобили, участвующие в них, двигаются со строго одинаковой скоростью и не способны поворачивать. Казалось бы, как тогда определить победителя? Все очень просто.

Из n стартовых точек, расположенных на одной стартовой прямой на равных расстояниях друг от друга, стартуют n автогонщиков, последовательно пронумерованных от 1 до n. Изначально каждый гонщик выставляет свою машину под некоторым углом к стартовой прямой. В момент старта все автомобили одновременно начинают движение под установленным углом, никогда не поворачивая. Каждая машина оставляет за собой глубокую колею. Если другая машина по ходу движения наезжает на колею, то застревает там навсегда. Победителями признаются все, кто проедет до финиша, не застряв в чужих колеях. Вы можете считать, что линия финиша расположена дальше самого позднего из теоретически возможных пересечений траекторий движения автомобилей.

Руководство федерации необычных гонок предполагает, что победителей гонки можно определить, просто зная угол старта каждого участника, сэкономив при этом кучу топлива.

Формат входного файла

Первая строка содержит одно целое число n ($2 \le n \le 10^5$) — количество автомобилей. Во второй строке через пробел записаны n действительных чисел $\alpha_1, \alpha_2, \ldots, \alpha_n$ ($0 < \alpha_i < \pi$), где α_i — угол наклона i-ой машины, заданный в радианах, не более чем с пятью знаками после десятичной точки. Углы измеряются от направления стартовой прямой, соответствующего увеличению номеров участников. Гарантируется, что никакие две машины не могут оказаться в одной точке одновременно.

Формат выходного файла

В первой строке выведите одно число — количество победителей гонки. Во второй строке выведите через пробел номера всех победителей в произвольном порядке.

Примеры

race.in	race.out
3 1.00 1.57 2.00	1 2
3 2.00 1.57 1.00	3 3 2 1

Задача M. Аптеки (Div-2 only!)

Имя входного файла:	`weights.in`
Имя выходного файла:	`weights.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

В городе Ангинске живут безупречно здоровые люди. И только один житель города постоянно недоволен этим фактом. Причина в том, что он работает аптекарем и вынужден постоянно сидеть без дела. Пытаясь хоть как-то развлечься, аптекарь придумывает математические задачки с аптечными гирьками и сам же их решает. Но одна оказалась на редкость сложной.

В аптеке используется набор из N гирь массой $1, 2^1, 2^2, \ldots, 2^{N-1}$ граммов и двухчашечные весы. Гири можно класть на обе чаши. Два способа взвешивания, отличающиеся только расположением чашек, считаются одинаковыми. Например, груз в 5 граммов с помощью трёх гирь $1, 2^1$ и 2^2 граммов можно взвесить двумя способами: $5 = 1 + 4$ и $5 + 1 = 2 + 4$.

Задача аптекаря состоит в том, чтобы найти количество способов взвешивания каждого товара в аптеке с помощью разных наборов гирь.

Формат входного файла

В первой строке содержится количество товаров T ($1 \le T \le 100$).

В каждой из последующих T строк записаны два натуральных числа N_i и M_i — число гирь, доступных для взвешивания товара, и вес этого товара соответственно($1 \le N_i < 63$, $1 \le M_i \le 5 \cdot 10^{18}$).

Формат выходного файла

Для каждого товара вывести количество возможных способов его взвешивания, по одному на строке.

Примеры

weights.in	weights.out
4 1 1 3 5 5 10 7 17	1 2 5 14

Задача N. Числа (Div-2 only!)

Имя входного файла:	`numbers.in`
Имя выходного файла:	`numbers.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Самое сложное испытание для студентов факультета занимательной математики — экзамен по цифровым сдвигам. Каждый год задачи по этой дисциплине становятся все сложнее, и к очередной сессии профессор Цифрский подготовил нечто невероятное.

Десятичная запись некоторого числа начинается с группы цифр, которую мы обозначим B. Другими словами, исходное число имеет вид BX. Если умножить это число на A, то результатом умножения станет число, получаемое переписыванием группы цифр B из начала в конец исходного числа, то есть итог умножения имеет вид XB.

Задача студентов — найти минимально возможное исходное число BX по заданным A и B.

Формат входного файла

Единственная строка содержит два целых числа A и B ($2 \le A \le 9$, $1 \le B \le 10^6$).

Формат выходного файла

Единственное число — решение задачи, либо `-1`, если решения не существует.

Примеры

numbers.in	numbers.out
5 1	-1
3 1	142857

Задача O. Прямоудартс (Div-2 only!)

Имя входного файла:	`darts.in`
Имя выходного файла:	`darts.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Все любят играть в дартс, но никому не нравится подсчет очков. Для обычного дартса этот процесс уже автоматизирован, но не для прямоугольного.

Прямоудартс играется на доске размером $n \times m$ с декартовой системой координат. Границы доски обведены черными линиями. Игрок, бросая каждый из k дротиков, гарантированно попадает в точку с целочисленными координатами в пределах доски. После каждого броска из точки попадания проводятся две черные линии, параллельные сторонам доски. По итогам игры за каждый квадрат с черными сторонами, найденный на доске, игроку начисляется по одному очку.

Федерация прямоудартса России объявила вознаграждение за алгоритм, который вычисляет количество набранных очков по координатам дротиков.

Формат входного файла

В первой строке — три целых числа n, m и k ($1 \le n, m \le 10^9$, $0 \le k \le 2000$).

В последующих k строках содержатся пары целых чисел x и y — координаты дротиков на доске ($0 \le x \le n$, $0 \le y \le m$).

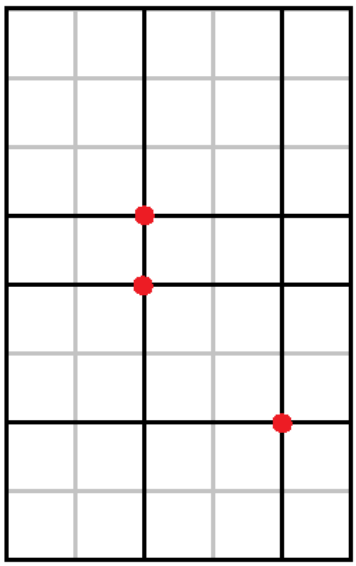
Формат выходного файла

Выведите одно число — количество набранных очков.

Примеры

darts.in	darts.out
5 8 4 4 2 2 4 2 5 4 2	10

Замечание



Задача P. Проволоки (Div-2 only!)

Имя входного файла:	`wire.in`
Имя выходного файла:	`wire.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Село Проволоки, расположенное недалеко от Казани, широко известно благодаря традиционным поделкам из металла — загогулинам, которые издревле изготавливаются проволокскими ремесленниками. Каждое подобное изделие создается на клетчатом столе, представляющем собой декартову систему координат. Мастер-загогульщик укладывает на стол длинный металлический прут, начиная с точки, имеющей целочисленные координаты. На каждом этапе мастер протягивает прут параллельно одной из координатных осей до другой точки с целочисленными координатами. Повторив эту операцию n раз, загогульщик фиксирует прут в очередной точке и отрезает излишки металла.

Известно, что загогулины чрезвычайно ценны благодаря своей аутентичности. Однако ценность каждой определяется довольно своеобразно. Каждый прямой изгиб загогулины оценивается в один рубль. Прямым считается сгиб прута под углом 90 градусов.

Профсоюз загогульщиков ждет появления программного продукта, который автоматизирует вычисление стоимости загогулины по её конфигурации.

Формат входного файла

В первой строке: $2 \le n \le 100$. В последующих n строках даны пары целых чисел x и y, не превосходящие по модулю 100, — координаты узлов изделия.

Формат выходного файла

Выведите одно число — цену загогулины в рублях.

Примеры

wire.in	wire.out
9 0 0 0 2 0 3 3 3 3 2 3 1 2 1 1 1 1 4	4
6 1 1 1 5 1 2 1 4 1 3 5 3	1