Задача A. Две корзины

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Аня и Боря играют в игру с двумя изначально пустыми корзинами. Они ходят по очереди, первой ходит Аня.

На каждом своём ходу Аня докладывает n яблок в корзины, при этом она может выбирать, сколько из этих n яблок она положит в первую корзину, а сколько — во вторую. Резать яблоки запрещено, поэтому оба числа должны быть целыми.

Боря же на своём ходу может выбрать любую корзину и съесть все яблоки из неё.

Игра продолжается неограниченно долго. Какое наибольшее количество яблок может быть в одной из корзин после хода Бори, если Аня стремится максимизировать это число, а Боря — минимизировать?

Более формально, можно доказать существование такого целого числа C, что Боря может добиться того, что после каждого его хода в обеих корзинах будет не больше C яблок, а Аня может добиться того, что после какого-то хода Бори в одной из корзин будет ровно C яблок. От вас требуется найти число C.

Формат входных данных

В единственной строке дано целое число n ($1 \leqslant n \leqslant 10^9$) — сколько яблок Аня кладёт в корзины на каждом своём ходу.

Формат выходных данных

Выведите одно целое число — наибольшее количество яблок, которое может оказаться в одной корзине после хода Бори, при условии, что и Аня, и Боря играют оптимально.

Система оценки

Тесты к этой задаче состоят из трёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов, подходящих под ограничения этой группы.

В первой группе $1 \leqslant n \leqslant 5$. За прохождение тестов первой группы можно получить 30 баллов.

Во второй группе $1 \leqslant n \leqslant 1000$. За прохождение тестов второй группы можно получить 40 баллов.

В третьей группе нет дополнительных ограничений на n, то есть $1 \leqslant n \leqslant 10^9$. За прохождение тестов третьей группы можно получить 30 баллов.

Примеры

стандартный ввод	стандартный вывод
1	0
3	2

Пояснения к примерам

В первом примере Боря может поддерживать обе корзины пустыми после каждого своего хода, поэтому ответ — 0.

Во втором примере у Ани есть стратегия, которая заставит Борю оставить хотя бы два яблока в одной из корзин после своего хода.

Действительно, Аня на первом своём ходу может положить одно яблоко в первую корзину и два во вторую. Тогда Боря опустошит вторую корзину (иначе Аня сразу добилась своего), после чего Аня доложит одно яблоко в первую корзину и два — во вторую. Теперь в обеих корзинах по два яблока, и Аня добьётся своего, независимо от того, какую корзину опустошит Боря.

С другой стороны, можно доказать, что Боря может добиться того, что после каждого его хода в обеих корзинах не больше двух яблок.

Задача B. Восстановление шестиугольника

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Объясняя решение задачи, Денис нарисовал на доске правильный шестиугольник. Через некоторое время место на доске понадобилось для решения другой задачи. Андрей предложил отметить три точки на доске, шестиугольник стереть, а потом по этим трём точкам восстановить, когда он снова понадобится.

Помогите Андрею выбрать такие три точки, а потом восстановить по ним правильный шестиугольник. Можно выбирать любые точки, а не только вершины шестиугольника.

Протокол взаимодействия

В этой задаче ваше решение будет запущено на каждом тесте два раза.

При первом запуске решение получает шесть вершин правильного шестиугольника и сохраняет три точки на плоскости. В первой строке записано число $n = 6$. Следующие n строк содержат координаты вершин — по два вещественных числа, строго меньших 1000 по абсолютной величине и записанных ровно с тремя знаками после десятичной точки (настоящие координаты округлены к ближайшим числам, которые можно так записать). В первой группе тестов вершины даны в порядке обхода против часовой стрелки, а во второй группе тестов вершины могут быть даны в произвольном порядке. Гарантируется, что сторона шестиугольника имеет длину не меньше 1.

Выведите три строки с координатами точек, которые следует сохранить. Помните, что эти точки не обязаны быть вершинами шестиугольника. Координаты этих точек также должны быть строго меньше 1000 по абсолютной величине, а количество знаков после десятичной точки может быть произвольным.

При втором запуске решение получает три сохранённых точки и восстанавливает шестиугольник. В первой строке записано число $k = 3$. Следующие k строк содержат координаты сохранённых точек — по два вещественных числа, ровно те, которые решение вывело при первом запуске, в том же порядке, но записанные ровно с тремя знаками после десятичной точки (если знаков было больше, число округляется к ближайшему, которое можно так записать).

Выведите шесть строк с координатами вершин исходного многоугольника. Вершины должны следовать в порядке обхода — либо по часовой стрелке, либо против часовой стрелки — и отличаться от исходных не более чем на 0.01 по каждой координате.

Примеры

На каждом тесте входные данные при втором запуске зависят от того, что вывело решение при первом запуске.

Далее показаны два запуска какого-то решения на первом тесте.

стандартный ввод	стандартный вывод
6 10.000 0.000 5.000 8.660 -5.000 8.660 -10.000 -0.000 -5.000 -8.660 5.000 -8.660	-10.0000000000 0.0000000000 -5.0000000000 -8.6600000000 10.0000000000 0.0000000000
3 -10.000 0.000 -5.000 -8.660 10.000 0.000	-10.0000000000 0.0000000000 -5.0000000000 -8.6600000000 5.0000000000 -8.6600000000 10.0000000000 0.0000000000 5.0000000000 8.6600000000 -5.0000000000 8.6600000000

Система оценки

Тесты к этой задаче состоят из двух групп. Баллы за первую группу (50) ставятся только при прохождении всех тестов этой группы и примера. Баллы за вторую группу (ещё 50) ставятся только при прохождении вообще всех тестов.

В первой группе (и в примере) гарантируется, что вершины изначально даны в порядке обхода против часовой стрелки. Во второй группе изначальный порядок вершин может быть произвольным.

Задача C. Домашнее задание

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

В университете Диме задали домой целых n задач по дискретной математике! Придя домой, он быстро все их сделал и сел решать задачи из архива на Codeforces. За задачами на динамическое программирование время пролетело незаметно, и, когда он устал и решил отдохнуть, была уже глубокая ночь.

Зайдя в чат одногруппников, Дима увидел сообщения о том, что преподаватель решил подшутить над студентами, и все правильные ответы почему-то содержат одну и ту же подстроку t. Перерешивать все задачи уже нет времени, ведь завтра утром их нужно сдавать, поэтому Дима решил просто подогнать все ответы под правильные. Так как Дима решил все задачи, то гарантируется, что каждый полученный им при решении ответ содержит хотя бы одну букву.

Дима не хочет ничего зачёркивать и исправлять, но может незаметно вставить в любые места ответа сколько угодно символов.

Так как времени очень мало, Дима хочет подогнать все ответы, добавив как можно меньше букв. Какое минимальное количество букв ему потребуется добавить суммарно во все ответы, чтобы каждый из них содержал подстроку t?

Формат входных данных

Первая строка входных данных содержит непустую строку t длины не более 500, состоящую из маленьких букв английского алфавита — подстроку, которая должна содержаться во всех правильных ответах.

Следующая строка содержит одно целое положительное число n — количество задач.

Последующие n строк содержат непустые строки s_i суммарной длины не более 10^4, также состоящие из маленьких английских букв — это ответы на задачи, которые получил при решении Дима.

Формат выходных данных

Выведите одно число — ответ на задачу.

Система оценки

Тесты к этой задаче состоят из трёх групп. Для каждой группы обязательно прохождение примера из условия и всех предыдущих групп тестов.

В первой группе $n = 1$, а длины строк t и s_1 не превосходят 10. За прохождение тестов первой группы можно получить 20 баллов.

Во второй группе длина строки t не превосходит 100, и суммарная длина строк s_i также не превосходит 100. Количество задач n не превосходит суммы длин строк s_i. Вторая группа оценивается в 40 баллов.

В третьей группе тестов длина строки t не превосходит 500, а суммарная длина строк s_i не превосходит 10^4. Количество задач n не превосходит суммы длин строк s_i. Третья группа также оценивается в 40 баллов.

Пример

стандартный ввод	стандартный вывод
prank 6 kotehok redpanda abcprankdef kaban geege burunduk	18

Пояснение к примеру

В первом примере можно:

- из первой строки добавлением четырёх символов получить «`kotehoPRANk`» (здесь и далее заглавные буквы — те, что были добавлены);
- из второй строки добавлением двух символов получить «`redpRanKda`»;
- с третьей строкой ничего не делать, так как она уже содержит подстроку «`prank`»;
- из четвёртой строки добавлением трёх букв получить «`kabPRanK`»;
- в пятую строку добавить все пять букв и получить «`PRANKgeege`»;
- из шестой строки добавлением четырёх букв получить «`buruPRAnKduk`».

Задача D. Жизнь на Марсе

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Майор Том бороздил Марс в поисках жизни, когда марсоход начал его подводить — система питания почти вышла из строя! К счастью, в этот момент на Красной планете бушевала магнитная буря.

Для простоты представим Марс как прямоугольное клетчатое поле размера $n \times m$. В каждой клетке определен некоторый вектор с целыми координатами, не превосходящими по модулю 1 — на такой вектор солнечный ветер сносит марсоход из этой клетки. Ветер, в том числе, может сносить марсоход за пределы Марса, но нельзя этого допустить — вернуться обратно невозможно, и, оказавшись там однажды, майору Тому придется вечно странствовать в открытом космосе.

Несмотря на неисправность, двигатели марсохода всё ещё работают и могут развивать любую целую мощность по любой координате. Вектор мощности суммируется с вектором магнитного поля в клетке, в которой находится марсоход, и он перемещается на получившийся вектор. Пусть, например, майор Том находится в клетке (i, j) с силой поля (r_{ij}, c_{ij}) и включил двигатель на вектор (u, v). Тогда в следующий момент времени он окажется в клетке с координатами $(i + r_{ij} + u, j + c_{ij} + v)$, после чего двигатель будет снова выключен. Но для того, чтобы развить мощность (u, v), потребуется $|u| + |v|$ единиц заряда.

Майор Том вызывает Землю и просит сказать, какое минимальное количество заряда потребуется, чтобы добраться до базы.

Формат входных данных

Первая строка входных данных содержит два целых числа n и m — количество строк и столбцов соответственно ($1 \leqslant n, m \leqslant 1000$).

Вторая строка содержит четыре целых числа — координаты майора Тома и базы: A_r, A_c, B_r, B_c ($1 \leqslant A_r, B_r \leqslant n$; $1 \leqslant A_c, B_c \leqslant m$).

Каждая из последующих n строк содержит по m пар целых чисел (r_{ij}, c_{ij}) — векторы поля в каждой клетке ($-1 \leqslant r_{ij}, c_{ij} \leqslant 1$).

Формат выходных данных

Выведите одно число — минимальное количество заряда, которое потребуется, чтобы добраться из начальной точки в конечную.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов этой группы, всех тестов всех необходимых групп, а также всех примеров.

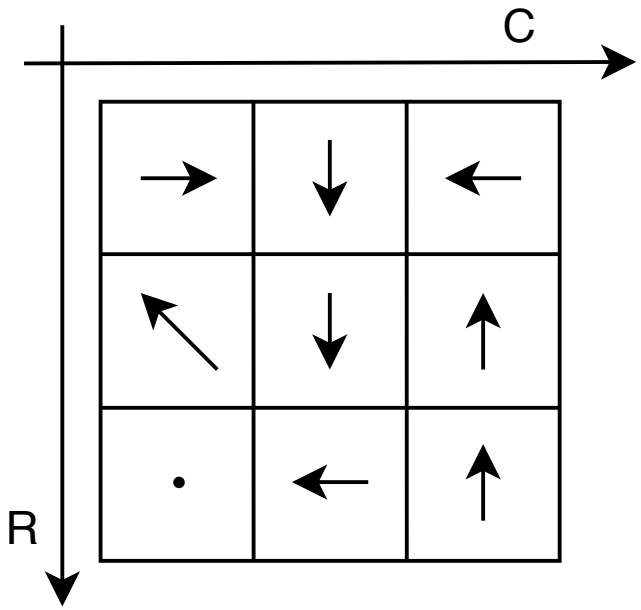
Группа	Баллы	Дополнительные ограничения	Необходимые группы	Комментарий
0	0	—	—	тесты из условия
1	10	$m \leqslant 1000$, $n = 1$, $r_{ij} = 0$	0	—
2	10	$m \leqslant 1000$, $n = 1$	0, 1	—
3	20	$n, m \leqslant 50$	0	—
4	60	$n, m \leqslant 1000$	0, 1, 2, 3	—

Примеры

стандартный ввод	стандартный вывод
3 3 1 1 3 3 0 1 1 0 0 -1 -1 -1 1 0 -1 0 0 0 0 -1 -1 0	1
3 5 1 1 2 5 0 -1 0 -1 0 -1 0 -1 0 -1 0 -1 0 -1 0 -1 0 -1 0 -1 0 1 0 1 0 1 0 1 0 1	4

Пояснения к примерам

В первом примере можно один раз включить двигатель на вектор $(0, 1)$ в клетке $(2, 2)$.



Задача E. А ну повтори!

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Это интерактивная задача.

В музее вы увидели постмодернистскую картину «Птичка в тумане». Ни птицу, ни туман вы на ней не разобрали, да и вообще картина представляла собой таблицу $n \times m$, в которой небольшое число клеток было чёрным, а остальные были белыми. Билет в музей был недешёвым, а то, что вложил художник в своё творение, вам так и не удалось до конца понять, поэтому вы запомнили, как выглядит картина, чтобы дома её воспроизвести и окончательно познать её смысл.

Дома вы развернули на всю стену холст, который можно считать бесконечной белой клетчатой плоскостью. Каждый день вы выбираете ровно две белые клетки и красите их в чёрный цвет: на то, чтобы красить больше, у вас нет сил и времени, а если красить меньше, то вас покидает неуловимый дух вдохновения. Ваша заветная мечта — в какой-то день проснуться и увидеть на холсте подпрямоугольник $n \times m$, который выглядит ровно как музейная картина и ориентирован так же: формально, в нём должны быть чёрными те и только те клетки (i, j), которые являются чёрными на картине. Найдя такой прямоугольник, вы не будете в этот день ничего закрашивать, а будете весь день просто созерцать этот прямоугольник и обретёте гармонию.

Но вот незадача — в доме напротив живёт снайпер-любитель Микола Поросюк. Вообще говоря, человек он хороший, вот только он решил тренировать своё мастерство на вашем холсте. Каждую ночь он целится из своей винтовки «Кабан-105» в одну из чёрных клеток и простреливает её. Вы уже привыкли за долгие годы невольного соседства к причудам Миколы, однако он сильно портит вашу затею: ведь, даже если вы в какой-то день приготовите нужный подпрямоугольник холста, вполне возможно, что этой же ночью Микола прострелит одну из его клеток. Любой прямоугольник, в клетке которого есть дыра, безнадёжно испорчен и не подойдёт для созерцания.

Ваш знакомый программист намекнул, что добиться своего вы всё равно сможете, если постараетесь. Удачи!

Протокол взаимодействия

В первой строке находится три целых числа n, m и B, разделённые пробелами — высота и ширина картины, а также число чёрных клеток на ней ($1 \leqslant n, m \leqslant 300\,000$, $1 \leqslant B \leqslant 13$). В следующих B строках находятся различные пары целых чисел p_i, q_i, разделённых пробелами — номер строки и столбца, содержащих i-ю чёрную клетку картины ($1 \leqslant p_i \leqslant n$, $1 \leqslant q_i \leqslant m$). Все заданные чёрные клетки различны.

Дальнейшее взаимодействие вашей программы с проверяющей системой состоит из нескольких запросов одного из двух видов.

1. Если в i-й день вы хотите покрасить две клетки в чёрный цвет, выведите отдельную строку в формате «`?` r_{2i-1} c_{2i-1} r_{2i} c_{2i}» — номера строк и столбцов нужных клеток (r_{2i-1}, c_{2i-1}) и (r_{2i}, c_{2i}). Эти две клетки должны отличаться друг от друга и ото всех клеток $(r_1, c_1), (r_2, c_2), \ldots, (r_{2i-2}, c_{2i-2})$, покрашенных на предыдущих ходах. После этого считайте из отдельной строки пару чисел s_i, t_i через пробел — номер строки и столбца чёрной клетки, которую i-й ночью прострелит Микола. Клетки (s_i, t_i) могут совпадать в разные дни.

2. Если в i-й день вы хотите созерцать готовый прямоугольник, выведите отдельную строку в формате «`!` r c» — номер строки и столбца верхнего левого угла вашего прямоугольника. Прямоугольник, строки которого имеют номера $r, r+1, \ldots, r+n-1$, а столбцы — $c, c+1, \ldots, c+m-1$, должен содержать ровно те чёрные клетки, которые соответствуют чёрным клеткам на картине, и в нём ни одна клетка не должна быть продырявлена Миколой в ночь с первой по $(i-1)$-ю.

Ваша программа должна сделать не более 8191 запросов первого типа, после чего она должна выполнить запрос второго типа и сразу завершиться. Все координаты клеток, которые она выводит,

должны быть целыми в пределах от -10^{18} до 10^{18} включительно.

После каждой выведенной строки следует очищать буфер вывода: это можно сделать, например, вызовом `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python.

Система оценки

Тесты к этой задаче состоят из семи групп. Баллы за каждую группу ставятся только при прохождении всех тестов этой группы, всех тестов всех необходимых групп и **первого примера**.

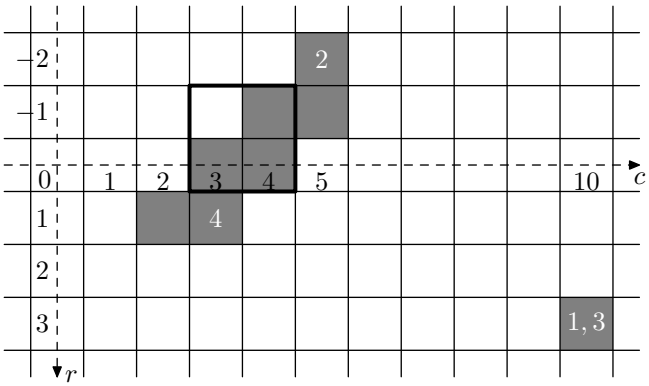
Группа	Баллы	n, m	B	Необх. группы	Комментарий
0	0	—	—	—	тесты из условия
1	6	$1 \leqslant n \cdot m \leqslant 10$	$B = 1$	—	—
2	6	$1 \leqslant n \cdot m \leqslant 10$	$1 \leqslant B \leqslant 2$	1	—
3	12	$1 \leqslant n \cdot m \leqslant 200$	$1 \leqslant B \leqslant 3$	0–2	—
4	21	$1 \leqslant n, m \leqslant 200$	$1 \leqslant B \leqslant 6$	0–3	—
5	18	$1 \leqslant n, m \leqslant 200$	$1 \leqslant B \leqslant 12$	0–4	—
6	26	$1 \leqslant n \cdot m \leqslant 13$	$1 \leqslant B \leqslant 13$	0–2	—
7	11	$1 \leqslant n, m \leqslant 300\,000$	$1 \leqslant B \leqslant 13$	0–6	—

Примеры

стандартный ввод	стандартный вывод
1 1 1 1 1 1 1	 ? 1 1 1 2 ! 1 2
2 2 3 1 2 2 2 2 1 3 10 -2 5 3 10 1 3	 ? -2 5 3 10 ? -1 5 -1 4 ? 0 4 1 3 ? 0 3 1 2 ! -1 3

Пояснения к примерам

Состояние холста в конце второго примера и подпрямоугольник, образующий копию картины, представлены на рисунке. Белые цифры означают номер ночи, в которую прострелена клетка.


−2
−1
0
1
2
3
1
2
3
4
5
10
c
r
2
4
1, 3

Задача F. Сумма расстояний

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мебибайт

Посчитайте количество помеченных деревьев на n вершинах, у которых сумма попарных расстояний между вершинами равна i, для каждого i от 1 до m. Поскольку эти количества могут быть очень большими числами, выведите их по модулю простого числа $10^9 + 7$.

Формат входных данных

В единственной строке даны два числа n и m ($2 \leqslant n \leqslant 26$, $1 \leqslant m$) — количество вершин в дереве и наибольшая возможная сумма попарных расстояний между вершинами.

Гарантируется, что во всех тестах m равно наибольшей возможной сумме попарных расстояний между вершинами в дереве с n вершинами.

Можно доказать, что $m = \dfrac{n^3 - n}{6}$.

Формат выходных данных

Выведите m целых чисел через пробел — искомые количества деревьев по модулю $10^9 + 7$.

Система оценки

Каждый тест в этой задаче оценивается отдельно.

Примеры

стандартный ввод	стандартный вывод
2 1	1
3 4	0 0 0 3
4 10	0 0 0 0 0 0 0 0 0 4 12

Пояснения к примерам

В первом примере есть всего одно помеченное дерево на двух вершинах.

Во втором примере есть три помеченных дерева на трёх вершинах, отличающиеся только нумерацией вершин. В каждом из них есть две пары вершин на расстоянии 1 и одна пара вершин на расстоянии 2, то есть общая сумма расстояний — 4.

В третьем примере есть 12 деревьев, выглядящих как путь, и 4 дерева, выглядящих как ёж. Можно убедиться, что в первых сумма попарных расстояний равна 10, а во вторых — 9.