,

Задача A. Угадай число

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 МБ

Это интерактивная задача. В процессе тестирования программа участника взаимодействует с программой жюри с использованием стандартных потоков ввода/вывода.

Программа жюри загадала число от 1 до n, цель программы участника — отгадать его, задав не более чем 30 вопросов. Для этого программа участника сообщает свои догадки программе жюри, а программа жюри отвечает, является ли загаданное число бóльшим, меньшим или равным сделанной догадке.

Протокол взаимодействия

Сначала необходимо прочитать из стандартного потока ввода число n ($1 \leqslant n \leqslant 10^9$). Затем протокол общения следующий: требуется вывести в стандартный поток вывода одну строку, содержащую целое число от 1 до n — свою догадку о загаданном числе.

После этого необходимо считать из стандартного потока ввода одно число: сообщение программы жюри. Возможны следующие сообщения:

- «`1`» — загаданное число больше последней догадки;
- «`-1`» — загаданное число меньше последней догадки;
- «`0`» — последняя догадка верна. Считав 0, ваша программа должна завершиться.

Обратите внимание на необходимость перевода строки после каждой выведенной догадки, прочитайте подробности про интерактивные задачи в памятке участника.

Пример

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

В примере ввод и вывод отформатированы пустыми строками, чтобы было видно, какие запросы соответствуют каким ответам программы жюри. В реальном взаимодействии необходимо переводить строку после каждого запроса, но выводить пустые строки не надо.

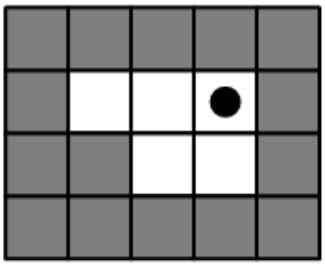
,

Задача B. Вслепую по лабиринту

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Ваша цель — написать программу, управляющую роботом, идущим вслепую по лабиринту.

Лабиринт состоит из n на m ($1 \leqslant n, m \leqslant 30$) клеток. Каждая из клеток может быть свободной или заблокированной (непроходимой). Все клетки на границе лабиринта непроходимые. Робот начинает работу в свободной клетке, он может переместится на юг, запад, север или восток в свободную клетку. Робот не имеет оптических сенсоров, только сенсор удара, так что при попытке перемещения в заблокированную клетке сработает сенсор и робот останется в той же клетке.



Робот должен побывать во всех проходимых клетках лабиринта. Из начальной клетки гарантированно можно попасть во все достижимые клетки лабиринта.

Формат входных данных

Каждая строка выходных данных должна представлять собой команду для робота. Это должна быть одна из пяти возможных строк: `SOUTH`, `WEST`, `NORTH`, `EAST` или `DONE`. Строка `DONE` должна быть напечатана после посещения роботом всех проходимых клеток. После вывода `DONE` программа должна завершать свою работу. Необходимо очищать выходной буфер после вывода каждой команды (`flush`).

Формат выходных данных

Каждая строка входных данных представляет собой ответ на действие робота. Это может быть строка `EMPTY`, если робот успешно переместился в заданном направлении, или строка `BLOCKED`, если робот не смог переместиться из-за того, что клетка, в которую он хотел попасть, непроходима.

Протокол взаимодействия

Программа должна выводить на стандартный вывод одну строку с действием робота и ждать строки в стандартном вводе с ответом, затем выводить очередную строку с действием и считывать ответ и так далее до тех пор, пока все проходимые клетки лабиринта не будут посещены. Программа должна завершать работу только когда все проходимые клетки будут посещены. Проходимые клетки могут быть посещены несколько раз. Допустимо передвигаться даже если все проходимые клетки уже посещены.

Пример

стандартный ввод	стандартный вывод
BLOCKED	NORTH
BLOCKED	EAST
EMPTY	SOUTH
BLOCKED	EAST
BLOCKED	SOUTH
EMPTY	WEST
BLOCKED	SOUTH
BLOCKED	WEST
EMPTY	NORTH
EMPTY	WEST
BLOCKED	WEST
BLOCKED	NORTH
EMPTY	EAST
BLOCKED	NORTH
	DONE

Задача C. В поисках истины

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

После вывода очередной строки не забывайте очищать буфер потока вывода после каждого запроса. Для этого можно, например, воспользоваться командами fflush(stdout), cout.flush(), либо выполнять перевод строки при помощи endl в C++, system.out.flush() в Java, flush(output) в Pascal или sys.stdout.flush() в Python.

Сарком управлял совет Великих сквайров. В совете состояло пятеро саркитов, и самый богатый и влиятельный из них, сквайр Файф, взял на себя ответственность разобраться в истории космоаналитика. Волей случая к нему попал и Рик, который оказался космоаналитиком, а Селим Джунц и Людиган Эбл из посольства Трантора были готовы с ним сотрудничать. Сквайра интересовало одно — кто же мог совершить столь ужасное преступление?

Еще раньше Великому сквайру поступали анонимные письма, в которых сообщалось, что Флорина должна погибнуть. Шантажист требовал отдать значительную долю кыртовых полей Файфа за эту информацию. Сквайр подозревал, что если преступник смог похитить космоаналитика, прятать его целый год от властей и при этом шантажировать самого главного человека на всем Сарке, он тоже должен быть Великим сквайром. И больше всего подозрений вызывал у него сквайр Стин.

Тут неожиданно Рик вспомнил, что во время разговора с похитителем, тот сообщил размер его владений на Флорине — k квадратных километров. В архиве хранятся данные о площади земель s_i, контролируемых сквайром Стином, в n моментов времени. Файфу известно, что все s_i различны, а так же что до некоторого момента эти площади увеличивались, а потом начали убывать. Более формально, существует такое $1 \leqslant t \leqslant n$, что для любого $1 < i \leqslant t$ $s_{i-1} < s_i$ и для любого $t < j \leqslant n$ $s_{j-1} > s_j$. Чтобы подтвердить свою правоту, Великому сквайру нужно узнать, в какой момент времени площадь владений Стина в точности равнялась k, и он просит вас помочь ему. Вы можете по моменту времени i узнать размер владений сквайра Стина s_i. Небольшая сложность состоит в том, что времени у Файфа мало, а направление запроса в архив происходит достаточно долго. Поэтому у вас есть возможность задать не более 80 таких запросов. Сквайр не сомневается в своем в успехе, и гарантирует вам, что искомое s_i существует.

Протокол взаимодействия

Изначально вам заданы два числа n, k — количество записей о владениях Стина в архиве и значение площади, которое интересует Файфа ($2 \leqslant n \leqslant 2 \cdot 10^5, 0 \leqslant k \leqslant 10^9$). Далее ваша программа может делать запросы вида "? i", в качестве ответа на которые программа жюри будет выводить значения s_i. Все s_i - целые числа, $0 \leqslant s_i \leqslant 10^9$. Записи в архиве нумеруются с 1. Когда ваша программа будет готова дать ответ, она должна вывести "! i", где $s_i = k$, и завершиться. Если ваша программа сделает больше 80 запросов первого типа, решение получит вердикт "Wrong answer". Если программа не завершится после запроса второго типа или ответ на запрос второго типа будет неверным, решение также получит вердикт "Wrong answer".

Пример

стандартный ввод	стандартный вывод
5 3 2 8 10 1 3	? 5 ? 4 ? 3 ? 1 ? 2 ! 2

Замечание

В данном примере записи о владениях выглядели так: $1, 3, 10, 8, 2$.

,

Задача D. Фишки

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это интерактивная задача. Ваша программа будет взаимодействовать с программой жюри, используя стандартный ввод и вывод.

Программа жюри решила сыграть с вашей программой в игру. На доске $n \times n$ в двух различных клетках находятся две фишки. Ваша программа должна определить положение фишек. Для этого она можете пытаться двигать фишки, а программа жюри будет сообщать результаты передвижений.

За один ход можно выбрать фишку и попросить переместить её на одну клетку влево, вправо, вверх или вниз. Программа жюри сообщает результат перемещения — если клетка в выбранном направлении существует и свободна, то перемещение считается успешным и фишка перемещается в эту клетку. В противном случае перемещение считается неудачным и фишка остается на той же клетке.

Вы выигрываете, если после очередного хода можете назвать исходное положение фишек на доске. Ваша задача — выиграть не более чем за $6n$ ходов.

Введем на доске систему координат таким образом, что клетки имеют координаты $(1, 1), (1, 2), \ldots, (1, n), (2, 1), \ldots, (n, n)$. Команды для перемещения фишки кодируется латинской буквой следующим образом:

- «`U`» — переместиться с клетки (x, y) на клетку $(x, y + 1)$.
- «`D`» — переместиться с клетки (x, y) на клетку $(x, y - 1)$.
- «`R`» — переместиться с клетки (x, y) на клетку $(x + 1, y)$.
- «`L`» — переместиться с клетки (x, y) на клетку $(x - 1, y)$.

Протокол взаимодействия

В самом начале программа жюри сообщает вашей программе натуральное число n ($2 \leqslant n \leqslant 50$) — размер доски.

Далее ваша программа должна повторять следующие ходы, выводя в стандартный поток вывода соответствующее сообщение и переводя строку.

- Если ваша программа считает, что определила начальное положение фишек, следует вывести 5 чисел: «1 x_1 y_1 x_2 y_2» ($1 \leqslant x_1, y_1, x_2, y_2 \leqslant n$) — начальное положение первой фишки (x_1, y_1) и второй фишки (x_2, y_2), соответственно. После вывода этой команды ваша программа должна завершиться.
- Если ваша программа хочет попытаться переместить фишку, следует вывести строку «0 *id* *c*», где *id* — номер фишки, которую ваша программа хочет переместить (1 или 2), а символ *c* — направление движения.

После каждого перемещения программа жюри сообщает вашей программе результат попытки перемещения:

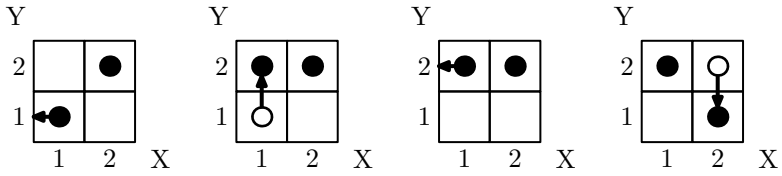
- «`1`», если передвижение успешное
- «`0`», если нет

Пример

стандартный ввод	стандартный вывод
2 0 1 0 1	0 1 L 0 1 U 0 1 L 0 2 D 1 1 1 2 2

Замечание

В примере фишки перемещались следующим образом.



В каждом тесте положение фишек исходно зафиксировано, и программа жюри честно отвечает на запросы вашей программы.

Задача E. Игра с бинарной строкой

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

Алиса и Боб играют в игру. У них есть строка длины n, каждый символ строки — это либо `0`, либо `1`. Игроки ходят по очереди, начинает Алиса. На каждом ходу игрок должен изменить **ровно** один символ строки: `0` меняется на `1`, а `1` — на `0`. После хода игрока должна получиться строка, которая раньше в игре никогда не встречалась (в том числе до всех ходов). Если игрок не может сделать ход, то он проигрывает.

Вам нужно выбрать, за кого (Алису или Боба) вы хотите играть, проверяющая система будет играть за другого игрока. Вы должны выиграть игру за выбранного игрока.

Протокол взаимодействия

В начале вы должны считать n ($1 \leqslant n \leqslant 15$) — длину строки — и строку s длины n — начальное состояние строки. После этого выведите «`Alice`» (если вы хотите играть за Алису) или «`Bob`» (если хотите играть за Боба).

В каждый свой ход вы должны вывести одно число p ($0 \leqslant p \leqslant n$).

- $p = 0$ символизирует, что вы сдаётесь. $1 \leqslant p \leqslant n$ — позиция символа, который вы меняет своим ходом. Позиции в строке нумеруются слева направо от 1 до n.

В каждый ход соперника вы должны считать одно число p ($-1 \leqslant p \leqslant n$).

- $p = 0$ символизирует, что соперник сдаётся. В этом случае вы должны завершить выполнение программы.
- $p = -1$ символизирует, что ваш последний ход привёл к строке, которая ранее встречалась, либо вы совершили недопустимый ход. В этом случае вы также должны завершить выполнение программы, иначе вы можете получить произвольный вердикт. $1 \leqslant p \leqslant n$ — позиция символа, который соперник меняет своим ходом.

Обратите внимание, что если вы выбрали Алису, то вы делаете первый ход, а если вы выбрали Боба, то вы делаете второй ход.

Не забудьте после каждого хода выполнять операцию '`flush`'.

Для сброса буфера вывода (то есть для операции '`flush`') сразу после вывода хода и перевода строки нужно сделать:

- fflush(stdout) в языке C++;
- System.out.flush() в Java;
- stdout.flush() в Python;
- flush(output) в Pascal;
- смотрите документацию для других языков.

В случае, если вы не будете выполнять операцию '`flush`' после каждого хода, либо не будете соблюдать формат взаимодействия с программой жюри, вы можете получить любой вердикт.

Пример

стандартный ввод	стандартный вывод
2 00 1 0	Alice 2 2

Задача F. Новогодний и прямоугольный

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Это **интерактивная** задача.

На Новый год Дед Мороз подарил Глебу то, о чём он уже давно мечтал — клетчатый квадрат размером $n \times n$. Подарок этот не простой, а с сюрпризом — внутри квадрата Дед Мороз выбрал некоторый непустой прямоугольник, и в каждую клетку этого прямоугольника он положил по мандарину.

Теперь, чтобы получить желанный подарок, Глебу нужно сыграть с Дедом Морозом в очень интересную игру. Глеб должен отгадать, в каком именно прямоугольнике находятся все мандаринки, подаренные Дедом Морозом. Будем считать, что строки и столбцы занумерованы числами от 1 до n снизу вверх и слева направо. Глеб может производить два типа запросов:

- ? $x_1\, y_1\, x_2\, y_2$ ($1 \leqslant x_1 \leqslant x_2 \leqslant n$, $1 \leqslant y_1 \leqslant y_2 \leqslant n$) — в ответ на этот запрос Дед Мороз говорит, сколько мандаринок находится в прямоугольнике, левым нижним углом которого является клетка (x_1, y_1), а правым верхним — клетка (x_2, y_2);
- ! $x_1\, y_1\, x_2\, y_2$ ($1 \leqslant x_1 \leqslant x_2 \leqslant n$, $1 \leqslant y_1 \leqslant y_2 \leqslant n$) — когда Глеб уверен, что он точно знает, где находятся мандаринки, он должен сделать запрос такого вида, чтобы сообщить свой ответ. При этом (x_1, y_1) соответствует предполагаемому расположению левого нижнего угла, а (x_2, y_2) — правого верхнего.

Формат входных данных

При запуске решения на вход вашей программе подается одно число n ($1 \leqslant n \leqslant 2 \cdot 10^9$) — размер квадрата.

Затем на каждый запрос типа "`?`" вам будет выдаваться количество мандаринок, находящихся в указанном вами прямоугольнике.

Формат выходных данных

Вы должны выводить корректные запросы в формате, описанном выше. Последним должен следовать единственный запрос вида "`!`", после чего ваша программа должна немедленно завершиться. Ваша программа должна произвести не больше 64 запросов типа "`?`". Обратите внимание, что последний запрос, выводящий ответ, не входит в данные 64 запросов.

В точности соблюдайте формат выходных данных. После вывода каждой строки сбрасывайте буфер вывода — для этого используйте команды `flush(output)` на языке Паскаль или Delphi, `fflush(stdout)` или `cout.flush()` в C/C++, `sys.stdout.flush()` на языке Python, `System.out.flush()` на языке Java.

Примеры

стандартный ввод	стандартный вывод
4 6 6 4	? 1 1 4 4 ? 1 3 4 4 ? 2 3 4 4 ! 1 3 3 4

Замечание

Пример в условии иллюстрирует взаимодействие с проверяющей программой. Для прохождения первого теста не обязательно производить такие же запросы, как в примере.

Задача G. Ехаб и еще одна очередная задача на xor

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это интерактивная задача!

Ехаб играет в игру с Лагги. Ехаб имеет два загаданных числа (a, b). Лагги может сказать пару чисел (c, d) и Ехаб ответит:

- 1 если $a \oplus c > b \oplus d$.
- 0 если $a \oplus c = b \oplus d$.
- -1 если $a \oplus c < b \oplus d$.

Операция $a \oplus b$ обозначает операцию побитовое исключающее «ИЛИ» чисел a и b.

Лагги нужно угадать (a, b) **не более, чем за 62 вопроса**. Вам предлагается сыграть в эту игру. Вы играете за Лагги, а программа жюри - за Ехаба.

Гарантируется, что $0 \leqslant a, b < 2^{30}$**.**

Формат входных данных

См. протокол взаимодействия.

Формат выходных данных

Чтобы вывести ответ, выведите "! a b"(без кавычек). **Не забудьте сбросить буфер вывода после того, как выведете ответ**.

Протокол взаимодействия

Чтобы задать вопрос, выведите "? c d"(без кавычек). c и d должны быть неотрицательными целыми числами, меньшими 2^{30}. **Не забудьте сбросить буфер вывода после того, как зададите вопрос**.

После каждого вопроса вы должны считать ответ. Если программа жюри отвечает числом -2, это значит, что ваша программа задала больше, чем 62 запроса и должна завершиться.

Чтобы сбросить буфер вывода вы можете использовать:

- fflush(stdout) в C++.
- System.out.flush() в Java.
- stdout.flush() в Python.
- flush(output) в Pascal.
- Прибегните к документации других языков.

Пример

стандартный ввод	стандартный вывод
1 -1 0	? 2 1 ? 1 2 ? 2 0 ! 3 1

Замечание

В примере из условия:

Загаданные числа: $a = 3$, $b = 1$.

В первом вопросе: $3 \oplus 2 = 1$ и $1 \oplus 1 = 0$, и ответ равен 1.

Во втором вопросе: $3 \oplus 1 = 2$ и $1 \oplus 2 = 3$, и ответ равен -1.

В третьем вопросе: $3 \oplus 2 = 1$ и $1 \oplus 0 = 1$, и ответ равен 0.

После этого программа выводит ответ и завершается.

,

Задача H. Сложная задача о печеньках

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это интерактивная задача. Существует 6 типов печенек. В середине дня номер i количество печенек типа j удваивается, если i делится нацело на j. Дни и типы печенек нумеруются с единицы.

Например, если изначально было 2 печеньки третьего типа, то к концу третьего дня их будет 4, а к концу шестого дня уже 8.

Вы можете сделать ровно два запроса общего количества печенек всех типов вечером дня i, после чего должны сказать сколько печенек каждого типа было изначально.

Поскольку печенек может оказаться очень много, интерактор будет сообщать количество печенек по модулю 2^{63}.

Гарантируется, что печенек каждого типа изначально было не более 100.

Протокол взаимодействия

Два раза повторяется следующая последовательность действий:

1. Ваша программа выводит одно число x — номер дня, вечером которого вы хотите узнать количество печенек, $1 \leqslant x \leqslant 500$.

2. Вашей программе подается на вход количество печенек вечером этого дня по модулю 2^{63}

После этого ваша программа должна вывести строку из 6 целых чисел – изначальные количества печенек каждого типа.

Не забывайте сбрасывать буфер вывода.

Пример

стандартный ввод	стандартный вывод
1 2	5 6 0 0 0 0 0 1

,

Задача I. Blindfolded Bullseye

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

У Гэри есть большая квадратная стена ровно 2×10^9 нанометров в высоту и 2×10^9 нанометров в ширину. У Гэри на стене висит доска для игры в дартс. Мишень для дротиков круглая, и ее радиус составляет от A до B нанометров включительно. Мишень для дротиков полностью находится внутри стены, но может касаться ее краев. Центр мишени-это целое число нанометров от каждого края стены.

Гэри пригласил своего друга Мику поиграть в интересную игру. Гэри завязывает Мике глаза и бросает ей вызов бросить дротик в центр мишени. Чтобы помочь ей, всякий раз, когда Мика бросает дротик в стену, Гэри скажет ей, попал ли дротик в мишень.

Мика не знает, где на стене доска для дротиков, но так как Мика очень искусна в дротиках, она может метать дротики с нанометровой точностью. То есть она может прицелиться и попасть точно в любую точку, находящуюся на расстоянии целого числа нанометров от каждого края стены. Сразу после броска каждого дротика Гэри говорит ей, попала ли она в центр мишени, в какую-то другую ее часть, или промахнулась совсем и ударилась о голую стену.

Можете ли вы помочь Мике попасть в центр мишени, не бросая более 300 дротиков?

Протокол взаимодействия

Первоначально ваша программа должна прочитать одну строку, содержащую два целых числа A и B ($5 \cdot 10^8 \leqslant A \leqslant B \leqslant 10^9$), обозначающие минимальное и максимальное значения радиуса мишени в нанометрах соответственно.

Мы представляем точки, в которые можно кидать дротики, как пары (x, y), где x и y — целые числа от -10^9 до 10^9 включительно. Пара (x, y) — это точка, которая находится на расстоянии $x + 10^9$ нанометров от левого края стены и $y + 10^9$ нанометров от нижнего края стены. Таким образом, точка $(0, 0)$ находится точно в центре стены.

Для каждого теста существует тайно выбранный радиус R для мишени и тайно выбранный центр мишени (X, Y). R, X и Y-целые числа, выбранные судьями для каждого теста. Для каждого тестового случая вам нужно сделать не более 300 запросов. Каждый запрос состоит из того, что ваша программа выбирает, куда бросить дротик, а интерактор дает информацию об этой позиции.

i-й запрос состоит из того, что ваша программа сначала выводит одну строку, содержащую два целых числа X_i и Y_i, оба между -10^9 и 10^9 включительно, а судья отвечает одной строкой, содержащей вердикт:

- `CENTER`, если $X_i = X$ и $Y_i = Y$
- `HIT`, если дротик попал в крут
- `MISS`, если дротик не попал в мишень

Сделав не более 300 запросов вам необходимо попасть в центр мишени.

Задача J. Медианная сила

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Во время проведения эксперимента в космосе вам довелось работать с N объектами, помеченными числами от 1 до N. известно, что N нечетно. Каждый объект обладает определенной, но неизвестной силой, выраженной натуральным числом. Для каждой силы Y выполняется, $1 \leqslant Y \leqslant N$. Объект с медианной силой — это такой объект X, что существует равное количество как объектов, имеющих силу, меньшую, чем X, так и объектов, имеющих большую силу, чем X. Вы должны написать программу, которая определяет объект с медианной силой. К сожалению, единственный способ сравнить силы — это использовать устройство, которое по трём различным объектам возвращает объект с медианной силой среди указанных трёх трех объектов.

Формат входных данных

Первая строка входных данных содержит одно целое число n — число объектов в эксперименте.
Ограничения:
Для числа N выполняется, что $5 \leqslant N \leqslant 1499$ и N нечетно.

- Для номеров объектов i выполняется $1 \leqslant i \leqslant N$.
- Для всякой силы Y выполняется $1 \leqslant Y \leqslant N$, и все силы различны.
- Разрешено сделать не более чем 7777 запросов определения медианы трёх объектов.

Формат выходных данных

Чтобы вывести ответ на задачу, выведите его в формате `! ans`, где ans — номер объекта с медианной силой.

Протокол взаимодействия

Чтобы задать запрос описанного в условии формата выведите `? a b c` ($1 \leqslant a \neq b \neq c \neq a \leqslant n$), где a, b, c — номера объектов, про которые вы хотите узнать информацию. Интерацирующая программа возвратит вам одно число — a, b или c.

Если возвращаемое значение равно -1, это означает, что вы превысили максимальное число запросов или сделали некорректный запрос. Ваша программа должна немедленно завершиться (например, вызовом `exit(0)`). Вы получите вердикт «Неправильный ответ», и это будет означать, что вы превысили максимальное число запросов или задали некорректный запрос. Если вы проигнорируете это, то можете получить любой вердикт, так как ваша программа продолжит читать из закрытого потока ввода.

Выше решение получит вердикт «Решение зависло», если вы не будете ничего выводить или забудете сделать операцию flush после вывода вопроса или ответа.

Чтобы выполнить операцию flush, можете использовать (сразу после вывода запроса и перевода строки):

- fflush(stdout) в C++;
- System.out.flush() в Java;
- stdout.flush() в Python;
- flush(output) в Pascal;

Для других языков смотрите документацию.

Пример

стандартный ввод	стандартный вывод
5 3 4 4 4	 ? 1 2 3 ? 1 2 4 ? 1 3 4 ? 4 2 5 ! 4

Замечание

Ниже изображено пояснение к примеру из условия:

Номер объекта	1	2	3	4	5
Сила объекта	2	5	4	3	1

Задача K. Силовое поле

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Империя обнаружила мятежников на ледяной планете Хот! По сведениям разведки все командование Альянса Повстанцев сейчас скрывается на базе «Эхо», спрятанной в горах на севере этой суровой планеты.

Для того, чтобы окончательно подавить силы восстания, необходимо в ходе стремительной атаки уничтожить эту базу и скрывающихся на ней мятежников. К сожалению, укрытие хорошо укреплено: в частности, его защищает мощное силовое поле, препятствующее бомбардировкам с орбиты. Силовое поле имеет форму выпуклого многоугольника с вершинами в N специальных станциях-ретрансляторах. Никакие три станции не располагаются на одной прямой.

Перед тем как начинать операцию по уничтожению повстанцев, требуется лишить их базу силового поля, уничтожив эти N станций точечным бомбометанием. Однако точные координаты этих станций нам неизвестны. Ваша цель — узнать расположение станций-ретрансляторов, чтобы наши войска смогли начать наступление.

На планете введена система координат, устроенная таким образом, что все станции-ретрансляторы находятся в точках с целыми координатами, не превосходящими C по модулю.

В вашем распоряжении есть зонд-разведчик, оснащенный специальным оборудованием, позволяющим регистрировать станции-ретрансляторы. Если запустить его по прямой над базой повстанцев, по его информации можно будет узнать, сколько станций-ретрансляторов располагаются слева, и сколько — справа от прямой его движения. Станции, находящиеся на его пути, зонд не регистрирует.

С повстанцами надо расправиться как можно скорее: у вас есть время не более чем на 10^5 запусков этого зонда. Восстановите по полученной от него информации точные координаты станций-ретрансляторов, чтобы мы могли начать наступление, и Империя вас не забудет!

Формат входных данных

Это интерактивная задача.

При запуске решения на вход подаются два целых числа N ($3 \leqslant N \leqslant 1\,000$) и C ($5 \leqslant C \leqslant 1\,000\,000$) — количество станций и ограничение на абсолютную величину их координат.

На каждый запуск зонда-разведчика вводится полученная им информация — два целых числа l и r, разделенных пробелом, — количество станций-ретрансляторов слева и справа от траектории его движения соответственно.

Формат выходных данных

Для запуска зонда выведите строку «? x_1 y_1 x_2 y_2», где (x_1, y_1) и (x_2, y_2) — две точки с целочисленными координатами, лежащие на прямой, по которой должен лететь зонд. Зонд будет лететь в направлении от первой точки ко второй. Точки не должны совпадать. Координаты точек не должны превосходить $5C$ по модулю.

Как только вы найдете ответ, выведите строку «Ready!», и в следующих N строках выведите координаты станций в любом порядке. После этого ваша программа должна завершиться.

Примеры

stdin	stdout
4 5 0 4 0 3 0 3 0 2 1 1 3 1 3 0 3 0	? -1 3 1 3 ? -1 2 1 2 ? -1 1 0 2 ? -1 0 0 2 ? 0 0 0 2 ? 1 0 1 2 ? 2 0 2 2 ? 3 0 1 2 Ready! 0 -1 2 1 0 2 -1 0

Замечание

В точности соблюдайте формат выходных данных. После вывода каждой строки сбрасывайте буфер вывода — для этого используйте `flush(output)` на языке Паскаль или Delphi, `fflush(stdout)` или `cout.flush()` в C/C++, `sys.stdout.flush()` на языке Python, `System.out.flush()` на языке Java.

Программа не должна делать более 10^5 запросов запуска зонда. При превышении этого количества, тест будет не пройден с вердиктом «Wrong Answer».

,

Задача L. Лошадь и Ежик

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это интерактивная задача.

Ёжик спустился в туман и оказался в прямоугольной долине размером N на M метров, по которой бродит Лошадь. Ёжик хочет ее найти. Будем считать, что в каждый момент времени и Ёжик, и Лошадь находятся в одной из $N \times M$ клеток. Туман настолько густой, что Лошадь не видно, даже если она находится в той же самой клетке , что и Ёжик. К счастью Ёжик обладает очень острым слухом и может понять, в каком направлении сместилась Лошадь. Он также может позвать Лошадь, и, если она находится в одной клетке с Ёжиком, то Лошадь его услышит и обязательно отзовется.

В каждый момент времени Ёжик может сместиться в соседнюю клетку по горизонтали, вертикали или диагонали. Потом он отчетливо слышит, куда сместилась Лошадь относительно своего старого местоположения. Лошадь за единицу времени смещается на одну клетку только по горизонтали или вертикали (влево, вверх, вправо или вниз). При этом Лошадь не выходит за границы долины, поэтому и Ёжик не должен этого делать.

Требуется написать программу, которая поможет Ёжику, знающему свое начальное положение и следящему за передвижениями Лошади, как можно быстрее ее найти. Кроме того, количество запросов о том, находится ли Лошадь в одной клетке с Ёжиком, не должно превышать $N \times M$.

Протокол взаимодействия

Сначала программа-решение должна прочитать из стандартного потока ввода натуральные числа N и M, записанные в первой строке, а из второй строки координаты начального местоположения Ёжика — два натуральных числа: x_0 — номер столбца, y_0 — номер строки ($1 \leqslant x_0 \leqslant M$, $1 \leqslant y_0 \leqslant N$). Числа в каждой строке разделены пробелом. Затем программа-решение начинает взаимодействие с программой, моделирующей поведение лошади, в соответствии со следующим протоколом:

1. Программа выводит в стандартный поток вывода одну строку, описывающую ход Ёжика, которая содержит три числа: его перемещение в виде указания смещения по горизонтали dx ($dx = -1,\ 0$ или 1) и по вертикали dy ($dy = -1,\ 0$ или 1), а также число 1, если Ёжик зовет Лошадь в клетке, в которую он при этом попадет, или 0 — если не зовет.

2. После этого программа должна считать из стандартного потока ввода ответ программы, сообщающей о действии Лошади. Ответ состоит из трех чисел, расположенных в одной строке через пробел. Первое число ответа может быть равно 0 или 1, где

   - 0 означает, что Ёжик не пытался позвать Лошадь либо позвал, но Лошади в его клетке нет. В этом случае следующие два числа обозначают очередное смещение Лошади по горизонтали dx ($dx = -1,\ 0$ или 1) и по вертикали dy ($dy = -1,\ 0$ или 1), при этом хотя бы одно из значений dx или dy равно нулю;
   - 1 означает, что Ёжик позвал Лошадь, и она действительно оказалась в той же клетке, что и он. В этом случае другие два числа равны 0, и программа-решение должна закончить свою работу.

Ваша программа должна сделать не более 10 000 ходов. Положительный вердикт на тесте вы получите, если выполнено следующее условие:

$$10 \cdot (\frac{J}{S})^2 \geqslant 9.5$$

Где J — это общее число ходов в программе жюри, а S — вашей программы. В данной задаче чекер адаптивный, и пытается играть с вами как можно дольше.

Пример

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

Замечание

Ёжик находился в клетке (1, 2). Сначала он попробовал позвать Лошадь в той же клетке (вывод: 0 0 1), но Лошади там не оказалось, и она сместилась вправо (ввод: 0 1 0). Ёжик сместился по диагонали, но Лошадь звать не стал (вывод: 1 –1 0), а Лошадь осталась на месте (ввод: 0 0 0). Ежик сместился вправо и позвал Лошадь (вывод: 1 0 1). Лошадь оказалась в той же клетке и отозвалась (ввод: 1 0 0). Значит, изначально Лошадь находилась в клетке (2, 1), а встретились они в клетке (3, 1). Ёжик при этом сделал три хода и дважды запросил местоположение Лошади.

Соблюдайте формат ввода-вывода, очищайте за собой буфер, и не теряйте друзей.