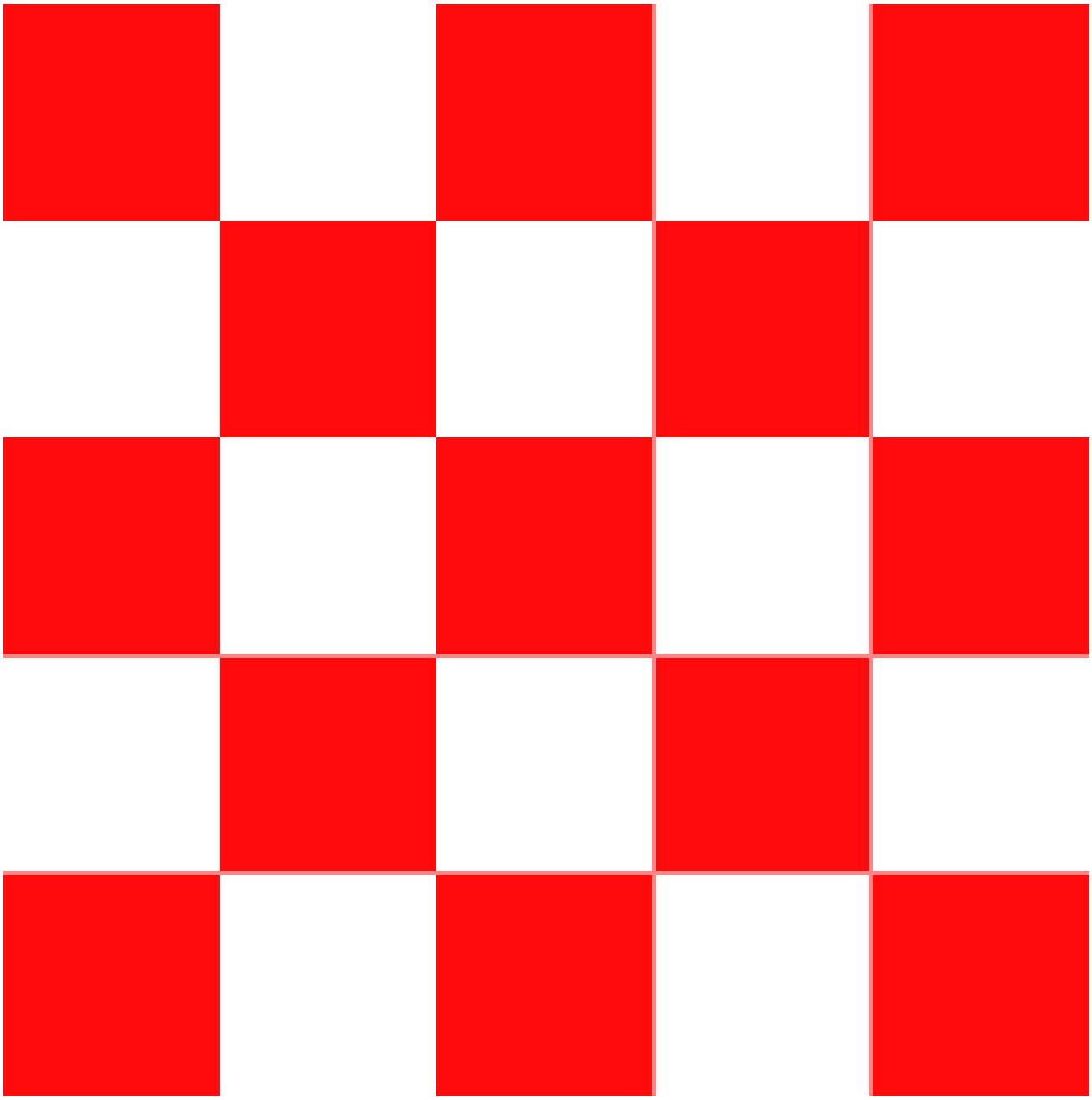
Задача A. Инопланетяне и Хорватия

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

Мирко большой фанат различных узоров на полях, в первую очередь странных кругов предположительно инопланетного происхождения. Одной летней ночью он решил создать свой собственный узор на поле своей бабушки. Так как Мирко патриот своей родной Хорватии, то он решил, что узор на поле будет в форме хорватского герба, который, как известно, представляет собой шахматную доску 5 на 5 с 13 красными и 12 бельми квадратами.



Поле бабушки Мирко разделено на N рядов по N клеток в каждом. Левый нижний угол поля обозначается координатами $(1, 1)$, правый верхний - координатами (N, N).

Мирко решил выкосить траву только на тех участках, которые соответствуют красным полям на шахматной доске. Он выбрал нечетное число $M \geqslant 3$ и так выкосил траву на поле, что каждый

квадрат на шахматной доске соответствует квадрату размером $M \times M$ клеток на поле, и шахматная доска целиком умещается на поле. На рисунке (см. английскую версию условия) показан пример поля для $N = 19$ и $M = 3$. Клетки, на которых трава была выкошена, отмечены серым. Центр узора имеет координаты (12,9) и отмечен черной точкой После того, как Мирко пошел спать, его творение привлекло внимание настоящих инопланетян! Они летают высоко вверху над полем в космическом корабле и исследуют узор с помощью прибора. Этот прибор может лишь определить, есть ли в определенной клетке трава или нет.

Пришельцы нашли одну клетку с выкошенной травой и теперь хотят найти центральную клетку узора Мирко. Они не знают размера узора M.

Протокол взаимодействия

Напишите программу, которая по размеру поля $N(1 \leqslant N \leqslant 2000000000)$, координатам некоторой клетки с выкошенной травой (X_0, Y_0) и способности взаимодействовать с инопланетным устройством, найдет центральную клетку узора Мирко

На каждом тесте устройство не может быть запущено более 300 раз

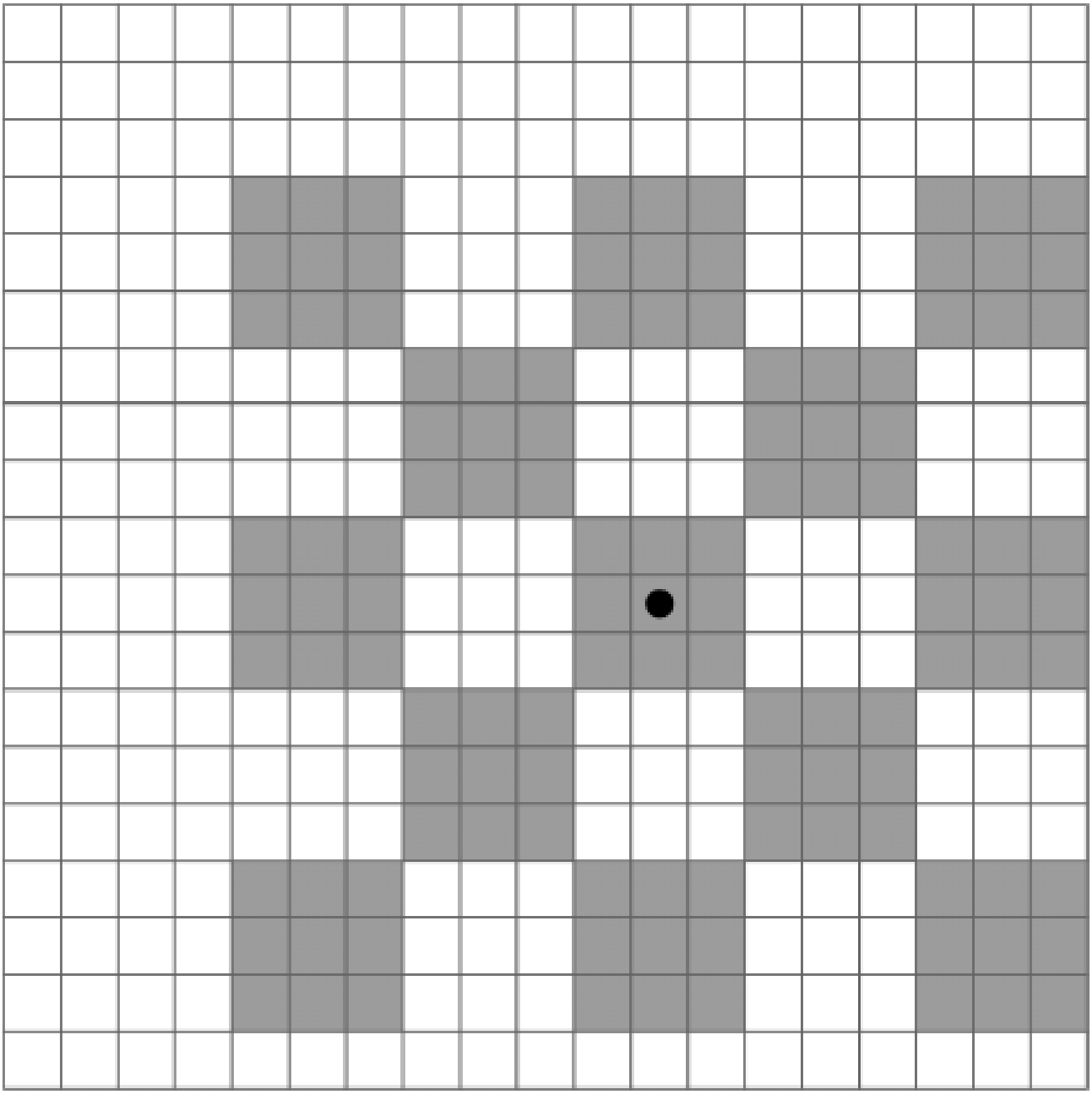
Это интерактивная задача. Ваша программа может взаимодействовать с устройством инопланетян, используя стандартный вывод и считывая данные, передаваемые устройством со стандартного ввода

- При запуске программы ей будут введены разделенные пробелом числа N, X_0, Y_0.

- Для того, чтобы узнать есть ли трава в клетке (X, Y) выведите на стандартный поток вывода строку "examine XY ". Если координаты (X, Y) не будут находиться внутри поля или вы запустите устройство более 300 раз, то космический корабль потеряет управление и врежется в землю. Вы же в этом случае получите 0 баллов за тест.

- Устройство поместит в ваш стандартный поток ввода строку "true если трава в указанной клетке выкошена и "false"в противном случае.

- Когда программа обнаружит центральную клетку, она должна вывести строку "solution $X_c Y_c''$ настандартный поток вывода. Выполнение вашей программы в этот момент будет автоматически завершено. Для корректной работы вашей программы не забывайте вызывать функцию "flush"после каждого вывода данных. Её заменяет endl в C++, print В Python, writeln В Pascal.



Замечание

Пример диалога
>20 4 9
< examine 2 9
> false
< examine 3 9
> true
< examine 6 9
> false
< examine 5 9
> true
< examine 4 3
> true
< examine 2 3
> false

< examine 3 3
> true
< examine 3 1
> false
< examine 3 2
> true
< solution 10 9

Задача B. 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 C. Простая 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 6 запросов 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

6 раз 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 1 1 1 1 2	1 2 3 4 5 6 0 0 0 0 0 1

Задача D. 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 E. 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 F. 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 G. Шляпа

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

Это интерактивная задача.

Имур Ихаков организует клуб по шляпе. На клуб пришло принять участие n человек, где n чётно. Имур усадил их всех в круг и провёл жеребьевку, чтобы разбить ребят на пары, но что-то пошло не так. Участники пронумерованы так, что участники i и $i+1$ ($1 \leqslant i \leqslant n-1$) сидят рядом, а также участники n и 1 сидят рядом. Каждому был выдан листочек с числом так, что у ребят, сидящих рядом, эти числа отличаются ровно на единицу. Предполагалось, что игроки с одинаковыми числами образуют пару, но оказалось, что не все числа встречались ровно дважды.

Как известно, удобнее всего объяснять слова партнёру, когда он сидит напротив. Участники с номерами i и $i + \frac{n}{2}$ сидят напротив друг друга. Имуру интересно, есть ли люди, сидящие напротив друг друга и имеющие одинаковые числа на своих листочках. Помогите ему найти такую пару людей, если она есть.

Вы можете задавать вопросы вида «какое число написано на листочке у школьника i?». Ваша цель — выяснить, есть ли требуемая пара, сделав не более 60 вопросов.

Формат входных данных

На вход подаётся одно целое чётное число n ($2 \leqslant n \leqslant 100\,000$) — число участников, пришедших на клуб Имура.

Вам разрешается задать не более 60 вопросов.

Формат выходных данных

Для того, чтобы узнать число i-го участника ($1 \leqslant i \leqslant n$), нужно вывести «`? i`». После этого ваша программа на вход получит целое число a_i ($-10^9 \leqslant a_i \leqslant 10^9$) — число на листочке i-го человека.

Чтобы сообщить, что ответ найден, требуется вывести «`! i`», где i — номер любого участника из пары ($1 \leqslant i \leqslant n$). Если такой пары участников не существует, выведите «`! -1`». После этого программа должна завершиться.

Запрос на вывод ответа не входит в ограничение на 60 запросов.

Не забывайте сбрасывать буфер после каждого запроса. Например, на C++ надо использовать функцию `fflush(stdout)`, на Java вызов `System.out.flush()`, на Pascal `flush(output)` и `stdout.flush()` для языка Python.

Взломы

Используйте следующий формат для взломов:

В первой строке выведите одно чётное целое число n ($2 \leqslant n \leqslant 100\,000$) — количество школьников на клуб Имура.

Во второй строке выведите n чисел a_i ($-10^9 \leqslant a_i \leqslant 10^9$) разделённые пробелами, где a_i — число, которое нужно написать на листочке i-му школьнику. Любые два соседних элемента, включая n и 1, должны отличаться на 1 или -1.

Взламываемое решение не будет иметь прямого доступа к последовательности a_i.

Примеры

стандартный ввод	стандартный вывод
8 2 2	 ? 4 ? 8 ! 4
6 1 2 3 2 1 0	 ? 1 ? 2 ? 3 ? 4 ? 5 ? 6 ! -1

Замечание

Ввод-вывод в примерах демонстрирует пример взаимодействия.

В первом примере был загаданы результаты жеребьёвки соответствующие последовательности $1, 2, 1, 2, 3, 4, 3, 2$

Во втором примере была загадана последовательность $1, 2, 3, 2, 1, 0$.

Задача H. Погоня в метро

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Это интерактивная задача. Параллельно с выполнением вашего решения жюри запускает проверяющую программу с который вы обмениваетесь сообщениями через стандартный ввод и вывод. Подробнее о протоколе взаимодействия написано ниже. Также в конце условия вы можете посмотреть корректные примеры взаимодействия с проверяющей программой на разных языках программирования.

В прекрасной Метрополии будущего необходимость в машинистах, управляющих поездами метро, отпала. Благодаря развитию технологий, их заменил искусственный интеллект (ИИ). К сожалению, в один прекрасный день опасения писателей-фантастов сбылись — ИИ взбунтовался, и теперь где-то в метро ездит неуправляемый поезд. Страшно представить, чем это грозит городской транспортной системе! Ваша задача состоит в том, чтобы найти поезд в сложной системе метро и остановить неуправляемый ИИ.

В целях безопасности все остальные поезда были отправлены в депо, а все ветки, кроме той, на которой находится неконтролируемый поезд, были перекрыты, поэтому на данный момент метро Метрополии представляет из себя одну ветку (обычную прямую ветку без самопересечений) из n станций, последовательно пронумерованных от 1 до n, ровно на одной из которых находится поезд. Для поимки неуправляемого поезда вам требуется определить номер этой станции, после чего на путях будут установлены искусственные заграждения и поезд будет пойман.

Для определения нужной станции диспетчер Сара одолжила вам устройство, позволяющее вам выбрать произвольные числа l и r ($l \leqslant r$), после чего оно проверит, верно ли, что поезд находится на станции с номером между l и r. К сожалению, для перезарядки устройства требуется k минут (и вы используете его как только перезарядка завершается), поэтому между двумя применениями поезд может перебраться из той станции, где он сейчас находится, в любую станцию с номером отличающимся не более чем на k. Формально, если при некотором применении устройства поезд находился на станции x, то при следующем применении он может находиться на любой станции y, такой что $\max(1, x-k) \leqslant y \leqslant \min(n, x+k)$. При этом поезд не знает, что вы пытаетесь его поймать и совершает все перемещения согласно некоторому заранее составленному им плану.

В процессе изучения устройства вы выяснили, что оно было сделано очень давно, и сможет выдержать не более чем q использований, после чего оно сломается, а ваша задача будет считаться проваленной.

Сможете ли вы найти станцию, на которой находится поезд, за не более чем q использований устройства?

Протокол взаимодействия

При запуске решения на вход подаются три целых числа n ($1 \leqslant n \leqslant 10^{18}$), k ($0 \leqslant k \leqslant 10$) и q ($q \geqslant 4\,500$). Ограничение на q следует читать следующим образом — чтобы решить задачу, ваша программа на любом тесте должна совершать не более 4500 использований устройства.

Чтобы использовать устройство, вы должны вывести через пробел два числа l и r ($1 \leqslant l \leqslant r \leqslant n$). В ответ на это вы получите либо строку «`Yes`», если между станциями с номерами l и r находится поезд, либо строку «`No`» иначе. Если $l = r$ и вы получили ответ «`Yes`», это значит, что вы точно определили станцию, на которой находится поезд, и ваша программа после этого должна немедленно завершиться.

Если ваша программа за q запросов не смогла точно определить станцию, на которой находится поезд, программа должна также немедленно завершиться, в противном случае вердикт тестирующей системы **может быть любым**.

После каждого запроса необходимо вывести перевод строки и сбросить 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.

В точности соблюдайте формат взаимодействия с системой.

Пример

стандартный ввод	стандартный вывод
10 2 15000 Yes No Yes	 3 4 3 3 2 2

Задача I. Из машины

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

Специальный агент Адам взламывает калькулятор члена секретной масонской организации иллюминатов, чтобы получить доступ к информации о её лидерах. Он уже выяснил, что всего в организацию входит n человек, каждый из которых имеет свой порядковый номер. Кроме того, у каждого из них есть уровень, представляющий собой целое число от 1 до n включительно. Уровни всех членов этой организации попарно отличаются. Любой иллюминат может командовать другими иллюминатами более низкого уровня, но вынужден подчиняться иллюминатам более высокого уровня. Адам полагает, что утечка информации об иллюминате наивысшего n-го уровня может его дискредитировать, поэтому ему интересен иллюминат с уровнем $n - 1$, подчиняющийся только иллюминату с наивысшим уровнем.

Доступ к секретной сети иллюминатов получить непросто, поэтому единственный вид запросов, которые может делать Адам — это запросы на сравнение уровней двух иллюминатов с заданными порядковыми номерами. В ответ на такой запрос он получает ответ, больше уровень первого иллюмината, чем уровень второго, меньше, или же их уровни равны. Количество запросов ограничено, но из-за особого отношения калькулятора иллюминатов к степеням двойки Адам может безопасно выполнить целых $(n + 24)$ запросов, прежде чем его присутствие в сети будет замечено.

Протокол взаимодействия

Это интерактивная задача. Ваша программа должна общаться с программой жюри, используя для этого стандартные потоки ввода и вывода.

Сначала в отдельной строке записано единственное целое число n $(2 \leqslant n \leqslant 1000)$ — количество иллюминатов.

После этого ваша программа может делать запросы на сравнение двух иллюминатов. Для этого в отдельной строке выведите символ «`?`» и два целых числа через пробел после него — порядковые номера иллюминатов, которых вы хотите сравнить. В ответ на это в отдельной строке будет записан единственный символ «`<`», если первый иллюминат имеет уровень меньше, чем второй, символ «`>`», если первый иллюминат имеет уровень больше, чем второй, либо «`=`», если указанные иллюминаты имеют одинаковый уровень. Вы можете сделать не более чем $(n + 24)$ таких запросов.

Когда вы узнаете ответ, выведите символ «`!`» и единственное целое число через пробел — номер иллюмината, имеющего уровень $n - 1$.

Пример

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

Замечание

Обратите внимание, что после вывода каждого сообщения ваша программа должна очищать потоковый буфер, чтобы выведенная вами информация дошла до программы жюри: например, это делают вызовы «`fflush(stdout)`» или «`cout.flush()`» в C++, «`System.out.flush()`» в Java, «`Console.Out.Flush()`» в C#, «`flush(output)`» в Pascal, «`sys.stdout.flush()`» в Python.