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 — 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. Одинокое число

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

Это интерактивная задача.

Однажды Маше было нечего делать, и она записала на листе бумаги N целых чисел: $a_1, a_2, \ldots, a_N$. Маша недавно изучила алгоритмы сортировки, поэтому она выписала свои числа в неубывающем порядке, то есть $a_1 \leqslant a_2 \leqslant \ldots \leqslant a_N$.

Также Маша очень любит загадки, поэтому среди ее чисел есть некоторое число C, которое встречается среди выписанных чисел ровно один раз, а все остальные числа встречаются ровно два раза.

Маша загадала вам загадку — найти «одинокое» число C. Для этого вы можете не более, чем 42 раза попросить Машу сообщить вам i-е записанное число.

Маша сообщила вам, что $1 \leqslant a_i \leqslant 10^9$.

Протокол взаимодействия

В начале ваше решение должно считать число N ($1 \leqslant N \leqslant 10^6$) — количество записанных Машей чисел.

Затем ваше решение может сделать не более 42 запроса. Для того, чтобы сделать запрос, ваше решение должно вывести его в следующем формате: «`?` i» (без кавычек, $1 \leqslant i \leqslant N$). Ответом на запрос является число a_i.

Для того, чтобы вывести ответ, ваше решение должно вывести «`!` C», после чего немедленно завершить работу.

Вы должны в точности соблюдать протокол взаимодействия с интерактором, в противном случае решение может получить произвольный вердикт.

При превышении числа запросов вы получите вердикт «`Неправильный ответ`».

После каждого запроса, в том числе после вывода ответа, вы должны выполнить операцию `flush`.

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

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

Если вы не сделаете операцию `flush` после какого-либо запроса, ваше решение может получить любой вердикт.

Пример

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

Замечание

В примере записанные числа равны: $1, 1, 4, 5, 5$. Одиноким числом, конечно, является число 4.

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. Угадай массив

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

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

Алиса и Боб решили сыграть в игру под названием «Угадай массив». Правила игры очень простые: Алиса загадала массив целых чисел размера n, а Бобу нужно угадать этот массив, сделав не больше n запросов о сумме на отрезке.

За один ход Боб может сделать к Алисе запрос одного из двух типов:

1. «`? l r`» — узнать сумму чисел на отрезке массива с l-го по r-й элемент включительно;
2. «`!`» — сообщить Алисе, что он готов дать ответ. После этого запроса Алиса ожидает от Боба n целых чисел — загаданный массив.

На каждый запрос первого типа, Алиса говорит Бобу сумму чисел на запрошенном отрезке. Но чтобы отгадывать было сложнее, после каждого запроса Алиса делает один отрезок *запрещенным*. В дальнейших запросах Боб не может запрашивать сумму чисел на запрещенных отрезках.

Помогите Бобу угадать загаданный массив, сделав не более n запросов первого типа.

Протокол взаимодействия

В первой строке ввода дано целое число n — размер загаданного массива ($1 \leqslant n \leqslant 10^4$). Гарантируется, что все числа в массиве по модулю не превосходят 10^9.

Далее запускается протокол взаимодействия с программой жюри — интерактором.

Интерактор ожидает от вашей программы запросы двух типов: «`? l r`» или «`!`», где l, r — целые числа — границы отрезка, на котором вы хотите узнать сумму ($1 \leqslant l \leqslant r \leqslant n$). После каждого запроса должен следовать перевод строки. При несоблюдении вашей программой формата запросов, ваше решение может получить произвольный вердикт (отличный от `OK`).

После запроса первого типа необходимо считать со стандартного ввода три целых числа s, l_b и r_b — сумму чисел на запрошенном отрезке s и границы нового *запрещенного* отрезка $[l_b, r_b]$, на котором запрашивать сумму больше нельзя ($|s| \leqslant 10^{18}$; $1 \leqslant l_b \leqslant r_b \leqslant n$).

Запрос второго типа означает, что ваша программа готова дать ответ на задачу. После запроса второго типа необходимо вывести n целых чисел — загаданный массив.

Обратите внимание, ваша программа может сделать не более n запросов первого типа. При превышении данного ограничения, а также при попытке узнать сумму на *запрещенном* ранее отрезке, интерактор выведет «`-1 -1 -1`» и завершится с вердиктом `WA`. Чтобы не получить вердикт `TL` или `IL`, считав из ввода значения «`-1 -1 -1`», ваша программа должна завершить свою работу с нулевым кодом возврата.

Пример

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

Задача E. Поиграем?

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

Это **интерактивная** задача.

1804 год. Вице-президент Соединённых Штатов Аарон Бёрр вызывает на дуэль кандидата в губернаторы штата Нью-Йорк Александра Гамильтона за серию оскорбительных памфлетов в свой адрес.

Но Бёрр разумный человек и понимает, что даже если он убьет Гамильтона на дуэли, он потеряет свою репутацию, и его политическая карьера будет закончена. Поэтому противники решили просто сыграть в игру. Для честности они решили сыграть в неё g раз.

Каждую игру Гамильтон загадывает целое положительное число n, а Бёрр пытается его отгадать. Для любого целого положительного x Бёрр может спросить у Гамильтона долю чисел между 1 и n включительно, которые делятся на x. Иными словами, спрашивая про x, он получает значение выражения

$$\frac{\left\lfloor \frac{n}{x} \right\rfloor}{n},$$

причём Гамильтон сообщает ему результат в виде **несократимой** дроби (здесь $\lfloor r \rfloor$ обозначает результат округления вниз вещественного числа r).

Помогите Бёрру найти ответ за некоторое заранее определённое число запросов.

Протокол взаимодействия

При запуске решения на вход подается целое число g — число игр между Гамильтоном и Бёрром ($1 \leqslant g \leqslant 1000$).

Для каждого теста зафиксировано число q ($6 \leqslant q \leqslant 60$) — максимальное количество запросов в одной игре. Гарантируется, что q запросов достаточно, чтобы решить задачу. Эти числа не сообщаются программе участника, но ограничения на эти числа в различных подзадачах приведены в таблице системы оценивания. Если программа участника делает более q запросов программе жюри, на этом тесте она получает в качестве результата тестирования «Wrong answer».

Чтобы сделать запрос, следует вывести строку «`X` t», где t — целое положительное число ($1 \leqslant t \leqslant 10^{18}$), для которого требуется узнать значение выражения

$$\frac{\left\lfloor \frac{n}{t} \right\rfloor}{n}$$

В ответ на каждый запрос программа получает через пробел два числа a и b — числитель и знаменатель этой дроби **после сокращения** — или число -1 в случае, если программа превысила ограничение по числу запросов.

Если программа определила загаданное число, то она должна вывести строку «`N` t», где t — предполагаемый ответ ($1 \leqslant t \leqslant 10^{18}$). Если ответ был правильный, то в ответ программа получает строку «`Correct`», а если неправильный, то она получает строку «`Wrong`».

После этого программа переходит к следующей игре, если они остались, иначе она должна завершиться.

Обратите внимание, в случае считывания числа -1 или строки «`Wrong`» вы **обязательно** должны сразу завершить вашу программу. В противном случае вердикт тестирующей системы может быть некорректным, в частности, вы можете получить вердикт Run-time error или Time limit exceeded!

Гарантируется, что в каждом тесте загаданные числа фиксированы в самом начале игры и не изменяются в зависимости от ваших запросов.

Система оценки

Тесты к этой задаче состоят из девяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

Группа	Баллы	Дополнительные ограничения
0	0	$q = 60$
1	30	$q = 60$
2	10	$q = 30$
3	4	$q = 20$
4	4	$q = 15$
5	5	$q = 10$
6	5	$q = 9$
7	11	$q = 8$
8	10	$q = 7$
9	21	$q = 6$

Пример

стандартный ввод	стандартный вывод
2 1 2 3 10 1 5 1 5 1 10 1 10 0 1 Correct 1 1 0 1 Correct	 X 2 X 3 X 5 X 4 X 6 X 10 X 11 N 10 X 1 X 2 N 1

Замечание

В первом примере $g = 2$. Приведены примеры запросов, по которым игрок угадывает, что в первой игре загадано число 10, а во второй 1. Эти же числа загаданы в первом тесте в тестирующей системе.

В точности соблюдайте формат выходных данных. После каждого вывода обязательно выводите один перевод строки и 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.

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 q (параметр зависит от номера группы) запросов типа "`?`". Обратите внимание, что последний запрос, выводящий ответ, не входит в данные q 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.

Тесты к этой задаче состоят из шести групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **предыдущих** групп.

Группа	Тесты	Баллы	Дополнительные ограничения		Комментарий
			n	q	
0	1 – 1	0	$n = 4$	$q = 1000$	Тест из условия.
1	2 – 12	10	$n \leqslant 10$	$q = 10\,000$	
2	13 – 23	20	$n \leqslant 100$	$q = 10\,000$	
3	24 – 34	20	$n \leqslant 10\,000$	$q = 20\,000$	
4	35 – 45	25	$n \leqslant 2 \cdot 10^9$	$q = 128$	
5	46 – ∞	25	$n \leqslant 2 \cdot 10^9$	$q = 64$	

Задача G. Угадай выражение

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	1024 мегабайта

Это интерактивная задача.

Жюри загадало некоторое арифметическое выражение, содержащее $n + 1$ переменных $a_0, a_1, \ldots, a_n$, которое вычисляется по формуле:

$$(\ldots(((a_0\ op_1\ a_1)\ op_2\ a_2)\ op_3\ a_3)\ldots\ op_n\ a_n) \bmod 10^9 + 7,$$

где $op_1, op_2, \ldots, op_n$ — это арифметические операции, каждая из которых является либо операцией «+» (сложение), либо операцией «×» (умножение), а $a_0, a_1, \ldots, a_n$ — переменные.

Например, если значения переменных равны $(a_0, a_1, a_2) = (1, 1, 2)$, а операции равны $(op_1, op_2) = (+, \times)$, то в результате вычисления выражения мы получим значение $((1 + 1) \times 2) \bmod 10^9 + 7 = 4$.

Вы можете несколько раз задать некоторые значения каждой из переменных и попросить жюри вычислить значение выражения. После этого вы должны угадать, чему равны $op_1, op_2, \ldots, op_n$.

Протокол взаимодействия

В начале взаимодействия с программой жюри вы должны считать целое число n ($1 \leqslant n \leqslant 4\,000$) — количество операторов в формуле.

После этого вы можете вывести не более 275 запросов вида: «? $a_0\ a_1\ \ldots\ a_n$» ($0 \leqslant a_i < 10^9 + 7$). После этого вы должны считать одно число — значение выражения для данных переменных.

Когда вы определите, чему равны $op_1, op_2, \ldots, op_n$, вы должны вывести запрос вида: «! s», где s — строка, состоящая из n символов `+` или `x` (маленькая латинская буква x), где i-й символ строки соответствует операции op_i.

Не забывайте выполнять операцию `flush` после каждого запроса (в том числе и после последнего).

В случае, если вы выведите больше 275 запросов на вычисление выражения, либо не будете соблюдать описанный протокол взаимодействия с программой жюри, вы можете получить любой вердикт.

Примеры

стандартный ввод	стандартный вывод
2 4 6	? 1 1 2 ? 1 1 3 ! +x
10 5 6224 640750	? 1 1 1 1 1 1 1 1 1 1 1 ? 0 4 2 4 2 4 2 4 2 4 2 ? 1 2 3 4 5 6 7 8 9 10 11 ! ++xxx+x+xx

Задача H. 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 I. Программирование квадрокоптеров

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

Школьники готовятся к участию в соревновании по программированию квадрокоптеров. Квадрокоптер, который используется в соревновании, может выполнять две команды: подняться вверх на 1 метр и опуститься вниз на 1 метр. Команда подъёма обозначается символом «`(`», а команда спуска — символом «`)`».

Программа для квадрокоптера представляет собой последовательность команд. Программа считается корректной, если, начав её исполнение на уровне земли и выполнив последовательно все команды, квадрокоптер снова оказывается на уровне земли. При этом в процессе выполнения программы квадрокоптер не должен пытаться опуститься ниже уровня земли.

Например, следующие программы являются корректными: «`()(())`», «`((()))`». Программа«`(((`» не является корректной, поскольку квадрокоптер завершает ее выполнение на высоте 3 метра над уровнем земли, программа «`())(`» также не является корректной, поскольку при выполнении третьей команды квадрокоптер пытается опуститься ниже уровня земли.

Участник соревнования написал корректную программу для квадрокоптера, состоящую из n команд, пронумерованных от 1 до n. Он загрузил её в память квадрокоптера для демонстрации во время соревнования. К сожалению, после загрузки программы в память квадрокоптера участник случайно удалил её на своём компьютере, а квадрокоптер не позволяет выгрузить программу из своей памяти.

К счастью, квадрокоптер поддерживает специальный режим отладки программы. В этом режиме квадрокоптер с загруженной в него программой может отвечать на специальные запросы. Каждый запрос представляет собой два целых числа: l и r, $1 \leqslant l \leqslant r \leqslant n$. В ответ на запрос квадрокоптер сообщает, является ли фрагмент загруженной в него программы, состоящий из команд с l-й по r-ю включительно, корректной программой для квадрокоптера, либо нет. Участник хочет с помощью режима отладки восстановить загруженную в квадрокоптер программу.

Требуется написать программу-решение, которая взаимодействует с программой жюри, моделирующей режим отладки квадрокоптера, и в итоге восстанавливает загруженную в квадрокоптер программу.

Протокол взаимодействия

Это интерактивная задача.

Сначала на вход подаётся целое число n — количество команд в программе квадрокоптера ($2 \leqslant n \leqslant 50\,000$).

Для каждого теста жюри зафиксировано число k — максимальное количество запросов. Гарантируется, что k запросов достаточно, чтобы решить задачу. Это число не сообщается программе-решению. Ограничения k в различных подзадачах приведены в таблице системы оценивания. Если программа-решение делает более k запросов к программе жюри, то на этом тесте она получает в качестве результата тестирования «Неверный ответ».

Чтобы сделать запрос, программа-решение должна вывести строку вида «`?` l r», где l и r — целые положительные числа, задающие фрагмент программы квадрокоптера ($1 \leqslant l \leqslant r \leqslant n$).

В ответ на запрос программы-решения программа жюри подаёт ей на вход либо строку «`Yes`», либо строку «`No`», в зависимости от того, является ли запрошенный фрагмент программы квадрокоптера корректной программой.

Если программа-решение определила ответ на задачу, то она должна вывести строку «`!` $c_1c_2\ldots c_n$», где символ c_i задаёт i-ю команду в программе квадрокоптера и равен либо «`(`», либо «`)`».

После этого программа-решение должна завершиться.

Гарантируется, что в каждом тесте программа в памяти квадрокоптера является фиксированной корректной программой, которая не меняется в зависимости от запросов, произведённых программой-решением.

Система оценки

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	k		
1	21	$2 \leqslant n \leqslant 16$	$k = 150$		Потестовые
2	28	$2 \leqslant n \leqslant 100$	$k = 10\,000$	1	Потестовые
3	26	$2 \leqslant n \leqslant 1000$	$k = 10\,000$	1, 2	Потестовые
4	25	$2 \leqslant n \leqslant 50\,000$	$k = 100\,000$	1 – 3	Потестовые

Примеры

стандартный ввод	стандартный вывод
4 Yes No Yes Yes	 ? 1 4 ? 1 3 ? 1 2 ? 3 4 ! ()()
6 Yes No Yes	 ? 3 4 ? 1 2 ? 2 5 ! ((()))

Замечание

Приведённые примеры иллюстрируют взаимодействие программы-решения с программой жюри «по шагам», для чего в них добавлены дополнительные пустые строки. При реальном тестировании лишние пустые строки вводиться не будут, выводить пустые строки также не требуется.

В первом примере $n = 4$. Единственная возможная корректная программа из двух команд это «`()`», поэтому из результатов третьего и четвёртого запросов можно сделать вывод, что программа в памяти квадрокоптера — «`()()`». Поэтому, в частности, ответ на второй запрос действительно «`No`», так как фрагмент программы «`()(`» не является корректной программой: если квадрокоптер исполнит именно эти три команды, он останется на уровне одного метра над землёй.

В втором примере $n = 6$, и произведённых запросов достаточно, чтобы однозначно определить, что программа в памяти квадрокоптера — «`((()))`».

В тестах из условия $k = 150$, то есть, разрешается произвести не более 150 запросов.

Задача J. 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 K. Последний рубеж

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

Это интерактивная задача.

Однажды общительный программист Павел позвал своих друзей на квест. Ребята с лёгкостью решали головоломки и продвигались вперёд. И вот им осталось решить последнюю загадку перед тем, как получить долгожданный приз.

Загадка состоит в том, что перед ребятами находится дверь с N замками, которую нужно открыть. Некоторые замки открыты, а некоторые закрыты. Ребята не знают, какие замки уже открыты, однако, потратив некоторое время на изучение одного конкретного замка, они могут определить, открыт он или нет. Рядом с дверью висит табличка, на которой написано, что самый левый замок открыт, а самый правый закрыт.

В процессе выполнения предыдущих заданий ребята выяснили, как открыть дверь. Пронумеруем замки слева направо числами от 1 до N. Тогда для того, чтобы открыть дверь, ребятам нужно найти замок с номером $i < N$, такой что замок i открыт, а замок $i + 1$ закрыт.

Как уже было сказано, для того, чтобы определить, является ли i-й замок открытым, им нужно подробно его осмотреть, потратив на это некоторое время. Так как у ребят осталось немного времени для выполнения последнего задания, они могут подробно осмотреть не более Q замков.

Помогите ребятам открыть дверь.

Протокол взаимодействия

В начале на вход программе подаётся одно целое число N ($2 \leqslant N \leqslant 10^{18}$).

После этого вы можете делать запросы вида `?` i, означающие, что ребята подробно осматривают замок с номером i. В ответ на подобный запрос вы получите число `0`, означающее, что замок закрыт, или число `1` в противном случае.

Если вы нашли ответ, вы должны сделать запрос вида `!` i, означающий, что вы считаете, что замок i открыт, а замок $i+1$ закрыт. После этого запроса вы должны завершить работу программы.

Считается, что в начале вы знаете состояние замков 1 и N.

Вы должны сделать не более Q запросов первого типа. В случае превышения этого ограничения ваше решение получит вердикт «Неправильный ответ».

В случае нарушения каких-либо правил взаимодействия с программой-интерактором, ваше решение может получить любой вердикт.

После каждого запроса, в том числе после запроса второго типа, вы должны выполнить операцию `flush`.

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

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

Если вы не сделаете операцию `flush` после какого-либо запроса, ваше решение может получить любой вердикт.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Оценка	Необходимые подзадачи
0	0	Тесты из условия	потестовая	—
1	20	$Q = 100$ $2 \leqslant N \leqslant 100$	подзадача	—
2	10	$Q = 100$ $2 \leqslant N \leqslant 10^5$ В начале закрыты ровно два замка	подзадача	—
3	30	$Q = 90$ $2 \leqslant N \leqslant 10^9$	подзадача	1, 2
4	20	$Q = 65$ $2 \leqslant N \leqslant 10^{18}$	подзадача	1, 2, 3
5	20	$Q = 60$ $2 \leqslant N \leqslant 10^{18}$	подзадача	1, 2, 3, 4

Для каждой подзадачи сообщаются набранные баллы, а также результат тестирования на первом непройденном тесте.

Примеры

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