Problem A. Последний рубеж

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	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.

Input

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

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.

Output

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

Examples

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

,

Problem B. A + B

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	2 секунды
Memory limit:	256 мегабайт

Это интерактивная задача. Ваше решение должно строго следовать описанному ниже протоколу взаимодействия с интерактором.

> «Вот бы все задачи в контестах были такие же простые, как “A + B”...»...

Выведите сумму чисел a и b.

Числа a и b загаданы интерактором и вам не сообщаются. Чтобы получить какую-то информацию про числа a и b, вы можете сообщить интерактору любое целое неотрицательное число x, после чего интерактор вернет вам число, равное

$$(a \,\&\, b \,\&\, x) + (a \,|\, b \,|\, x) + (a \oplus b \oplus x)$$

Здесь за &, | и $\oplus$ обозначены операции побитового «и» (`and`), «или» (`or`) и «исключающего или» (`xor`) соответственно. Таким образом, на каждый ваш запрос интерактор сообщает вам сумму трех величин, получаемых из a, b и вашего числа с помощью каждой из данных операций.

Сделав не более 5 запросов к интерактору, найдите и выведите значение величины $a + b$.

Input

Интерактор загадывает два целых неотрицательных числа a и b ($0 \leqslant a, b \leqslant 10^9$). На ввод вашей программе ничего не поступает.

Output

Ваша программа может посылать интерактору запросы в формате «`?` x» ($0 \leqslant x \leqslant 10^{12}$). В ответ на каждый запрос такого вида интерактор сообщает вашей программе число $(a \,\&\, b \,\&\, x) + (a \,|\, b \,|\, x) + (a \oplus b \oplus x)$.

Если переданный в запрое x не укладывается в указанные ограничения или если превышается ограничение в 5 запросов на один тест, интерактор выводит «`-1`» (без кавычек) и завершается, а ваше решение решение получает вердикт `WA` (Wrong Answer). Считав «`-1`», ваша программа должна завершиться, иначе может быть получен некорректный вердикт `TL` (Time Limit Exceeded).

Если ваша программа готова дать ответ на задачу, следует вывести запрос формата «`!` x», где x — предполагаемое значение суммы $a + b$. Считав такой запрос, интерактор выставляет решению вердикт `WA` или `OK` в зависимости от правильности ответа и завершается. Соответственно, после вывода такого запроса ваша программа тоже должна завершаться.

После каждого запроса необходимо сбрасывать буфер вывода (`cout.flush()` в C++, `sys.stdout.flush()` в Python) и выводить перевод строки. После каждого ответа на запрос интерактор также выводит перевод строки. Лимит в 5 запросов не учитывает запрос второго типа, то есть можно задать 5 запросов формата «`?` x» и 1 запрос формата «`!` x».

Examples

стандартный ввод	стандартный вывод
3	? 1 ! 2

Note

В примере из условия делается запрос для $x = 1$, на что интерактор сообщает, что $(a \,\&\, b \,\&\, 1) + (a \,|\, b \,|\, 1) + (a \oplus b \oplus 1) = 3$.

На основе этой информации решение делает предположение, что $a = b = 1$. Простым перебором всех a и b, для которых $a \,|\, b \,|\, 1 \leqslant 3$ можно доказать, что другие значения a и b не могут дать ту же сумму, поэтому решение сразу выдает $1 + 1 = 2$ в качестве ответа.

,

Problem C. Угадай длину цикла

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	256 мегабайт

Это интерактивная задача.

Я хочу сыграть с тобой в одну игру...

Мы загадали циклический граф из n вершин ($3 \le n \le 10^{18}$). Циклическим называется неориентированный граф из n вершин, образующих один цикл. Каждая вершина принадлежит циклу, то есть длина цикла (количество рёбер в нём) равна в точности n. Порядок вершин в цикле — произвольный.

Вы можете делать запросы следующего вида:

- «`? a b`», где $1 \le a, b \le 10^{18}$ и $a \ne b$. В ответ на запрос интерактор выведет в отдельной строке длину случайного из двух путей из вершины a в вершину b, либо `-1`, если $\max(a, b) > n$. Один из двух путей интерактор выбирает **равновероятно**. Длина пути — это количество рёбер в нём.

Вы победите, если угадаете количество вершин в загаданном цикле (число n), сделав не больше 50 запросов.

Обратите внимание, что интерактор реализован так, что для любой упорядоченной пары (a, b), на запрос «`? a b`» он всегда возвращает одно и то же значение, независимо от количества таких запросов. Отметим, что на запрос «`? b a`» интерактор может дать другой ответ.

Вершины в графе расставлены случайно и их позиции заранее фиксированы.

В этой задаче запрещены взломы. Количество тестов у жюри равно 50.

Interaction Protocol

Вы можете сделать не более 50 запросов. Чтобы сделать запрос, выведите в отдельной строке:

- «`? a b`», где $1 \le a, b \le 10^{18}$ и $a \ne b$. В ответ на запрос интерактор выведет в отдельной строке длину случайного простого пути из вершины a в вершину b (не путать с путем из b в a), либо -1, если $\max(a, b) > n$. Один из двух путей интерактор выбирает **равновероятно**.

Если в результате запроса вашей программой будет получено число `0`, то это означает, что вердикт для вашего решения уже определён как «*Неправильный ответ*» (например, вы сделали больше 50 запросов или совершили некорректный запрос). В этом случае ваша программа должна немедленно завершить свою работу. В противном случае в этом сценарии вы можете получить случайный вердикт «*Ошибка исполнения*», «*Превышено ограничение времени*» или какой-то другой вместо вердикта «*Неправильный ответ*».

Ответ, как и запросы, выведите в отдельной строке. Вывод ответа не считается запросом при подсчёте их количества. Чтобы вывести его, используйте следующий формат:

- «`! n`»: предполагаемую длину загаданного цикла ($3 \le n \le 10^{18}$).

После этого ваша программа должна завершить работу.

После вывода очередного запроса обязательно используйте функции очистки потока, чтобы часть вашего вывода не осталась в каком-нибудь буфере. Например, на C++ надо использовать функцию `fflush(stdout)`, на Java вызов `System.out.flush()`, на Pascal `flush(output)` и `stdout.flush()` для языка Python.

Обратите внимание, что интерактор реализован так, что для любой упорядоченной пары (a, b), на запрос «`? a b`» он всегда возвращает одно и то же значение, независимо от количества таких запросов. Отметим, что на запрос «`? b a`» интерактор может дать другой ответ.

Вершины в графе расставлены случайно и их позиции заранее фиксированы.

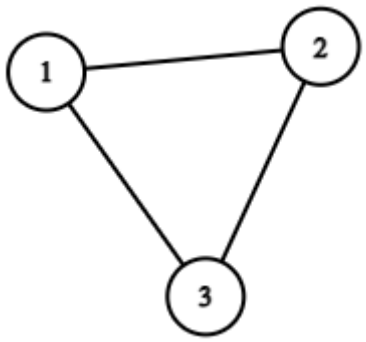
В этой задаче запрещены взломы. Количество тестов у жюри равно 50.

Example

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

Note

В первом примере цикл может выглядеть следующим образом:



Длины простых путей между всеми парами вершин в данном случае равны 1 или 2.

- Первым запросом мы узнаем, что один из простых путей из вершины 1 в вершину 2 имеет длину 1.
- Вторым запросом мы узнаем, что один из простых путей из вершины 1 в вершину 3 имеет длину 2.
- Третьим запросом мы узнаем, что вершины 4 нет в графе. Следовательно, размер графа равен 3.

Problem D. Программирование квадрокоптеров

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	2 секунды
Memory limit:	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, $1 \le l \le r \le n$. 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.

Input

Это интерактивная задача.

Сначала на вход подаётся целое число n — количество команд в программе квадрокоптера ($2 \le n \le 50000$).

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 ($1 \le l \le r \le n$).

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_1 c_2 ... 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.

В тестах $k = 100000$, то есть, разрешается произвести не более 100000 запросов.

Output

Examples

стандартный ввод	стандартный вывод
4 <ожидание ответа> Yes <ожидание ответа> No <ожидание ответа> Yes <ожидание ответа> Yes <ожидание ответа>	<ожидание ввода> ? 1 4 <ожидание ввода> ? 1 3 <ожидание ввода> ? 1 2 <ожидание ввода> ? 3 4 <ожидание ввода> ! ()()
6 <ожидание ответа> Yes <ожидание ответа> No <ожидание ответа> Yes <ожидание ответа>	<ожидание ввода> ? 3 4 <ожидание ввода> ? 1 2 <ожидание ввода> ? 2 5 <ожидание ввода> ! ((()))

,

Problem E. Интерактивная восточная сказка

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	2 секунды
Memory limit:	256 мегабайт

У злого волшебника Джафара много ламп, которые он холит и лелеет, и любит очень сильно, но из каждой пары ламп он всё же может выбрать одну, которую он любит даже сильнее, чем другую.

Он захотел расставить их в ряд так, чтобы когда он будет идти вдоль этого ряда каждая следующая лампа была им более любима, чем предыдущая.

Новому слуге Джафара поручено это сделать, но... он не знает предпочтений Джафара!

Про любую пару ламп можно спросить у волшебника, какую он любит больше, но нельзя излишне навязываться с вопросами (отрубание головы еще никто не отменял).

Помогите слуге расположить лампы или узнать, что это невозможно (и сброситься со скалы).

Input

Первое число, которое будет передано во входном потоке, — N ($1 \le N \le 1000$), количество ламп.

Затем на каждый вопрос слуги, который ваша программа выведет в выходной поток, во входном потоке будет дан ответ — слово "YES", если лампа Y_i более любима чем X_i, и слово "NO", если X_i более любима чем Y_i.

Заметьте, что отношение «более любима чем» не обязано быть транзитивным.

Output

В выходной файл вы можете выводить запросы Каждый вопрос — одна строчка с тремя числами 1, X_i, Y_i ($1 \le X_i, Y_i \le N$; $X_i \ne Y_i$). Вы можете задать не более 10 000 вопросов.

В последней строчке выведите число 0, а затем N целых числел от 1 до N — номера ламп в порядке, в котором их надо расставить. Если же расставить лампы невозможно, выведите $(N + 1)$ ноль.

Examples

стандартный ввод	стандартный вывод
3 YES NO	 1 1 2 1 1 3 0 3 1 2

Note

Обязательно сбрасывайте буфер при выводе запросы: в Pascal используйте `flush(output);`, в C используйте `fflush(stdout);`, в C++ используйте `cout.flush();`, в Java используйте `System.out.flush();`.

,

Problem F. Новогодний и прямоугольный

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	3 секунды
Memory limit:	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:

- ? $x_1\, y_1\, x_2\, y_2$ ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — 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);
- ! $x_1\, y_1\, x_2\, y_2$ ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — 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.

Input

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

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

Output

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.

Example

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

,

Note

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	$n \le 10$	$q = 10\,000$	
2	13 – 23	20	$n \le 100$	$q = 10\,000$	
3	24 – 34	20	$n \le 10\,000$	$q = 20\,000$	
4	35 – 45	25	$n \le 2 \cdot 10^9$	$q = 128$	
5	46 – ∞	25	$n \le 2 \cdot 10^9$	$q = 64$	

Problem G. Интеракдивная задача

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	1 секунда
Memory limit:	256 мегабайт

Это интерактивная задача.

Мы решили сыграть с вами в игру и загадали число x ($1 \le x < n$), где число n вам известно.

Вы можете делать запросы следующего вида:

- `+ c`: эта команда осуществляет присвоение $x = x + c$ ($1 \le c < n$), а затем возвращает вам значение $\lfloor \frac{x}{n} \rfloor$ (x делить на n и округлить вниз).

Вы победите, если угадаете текущее число, сделав не больше 10 запросов.

Interaction Protocol

Взаимодействие начинается с чтения целого числа n ($2 < n \le 1000$), которое записано во входных данных на отдельной строке.

Далее вы можете сделать не более 10 запросов. Чтобы сделать запрос выведите в отдельной строке:

- `+ c`: эта команда выполнит присвоение $x = x + c$ ($1 \le c < n$), а затем в отдельной строке выведет значение $\lfloor \frac{x}{n} \rfloor$ (x делить на n и округлить вниз).

Ответ, как и запросы, выведите в отдельной строке. Вывод ответа не считается запросом при подсчёте их количества. Чтобы вывести его используйте следующий формат:

- `! x`: текущее значение x.

После этого ваша программа должна завершить работу.

После вывода очередного запроса обязательно используйте функции очистки потока, чтобы часть вашего вывода не осталась в каком-нибудь буфере. Например, на C++ надо использовать функцию `fflush(stdout)`, на Java вызов `System.out.flush()`, на Pascal `flush(output)` и `stdout.flush()` для языка Python.

Обратите внимание, что интерактор не является адаптивным.

Чтобы сделать взлом, используйте следующий формат: в единственной строке должны содержаться два числа x и n, разделённые пробелом.

,

Examples

стандартный ввод	стандартный вывод
3 1	+ 1 ! 3
5 0 0 1	+ 1 + 1 + 1 ! 5
10 0 0 1 2	+ 2 + 2 + 3 + 8 ! 20

Note

В первом примере изначально $x = 2$. После первого запроса значение $x = 3$, а $\lfloor \frac{x}{n} \rfloor = 1$.

Во втором примере также изначально $x = 2$. После первого запроса значение $x = 3$, а $\lfloor \frac{x}{n} \rfloor = 0$. После второго — $x = 4$, $\lfloor \frac{x}{n} \rfloor = 0$. После третьего $x = 5$, $\lfloor \frac{x}{n} \rfloor = 1$.

,

Problem H. E-бюрократ

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мегабайт

Это интерактивная задача

Вы реализуете для корпорации «Геркулес» систему «E-бюрократ», которая принимает на вход заявки от различных отделов, ставит случайную резолюцию, подписывает их электронной подписью директора корпорации и возвращает назад.

К сожалению, разные отделы не придерживаются каких-то общих стандартов в наименовании заявок, хотя заявки от одного учреждения имеют один и тот же формат и в рамках этого формата всегда нумеруются последовательными целыми числами, начинающимися с единицы.

Более формально, формат названия заявки в каждом отделе — это строка длины не более, чем 80 символов, состоящая из строчных букв, цифр и **ровно одной** *форматной группы*. Форматная группа имеет или вид `%d`, или вид `%0`x`d`, где x — цифра от 2 до 9. В первом случае номер заявки выводится вообще без ведущих нулей, во втором случае, если номер заявки имеет длину менее x знаков, то выводятся ведущие нули так, чтобы длина номера была ровно x, в противном случае ведущие нули не выводятся. Например, если формат названия — `berlaga1bukh%02d`, то заявка 1 будет называться `berlaga1bukh01`, заявка 42 — `berlaga1bukh42`, заявка 322 — `berlaga1bukh322`.

Заметим, что форматная группа может быть как в начале названия файла, так и в конце, а также следовать после цифр (например, форматы `%dzayavka`, `zayavka%09d`, `agent007%02drequest` являются корректными).

Вы хотите узнать формат для конкретного отдела. Для этого вы можете запросить, как будет называться заявка с номером x, где x — произвольное целое число от 1 до 10^9-1. Чтобы не отвлекать сотрудников от важной работы по написанию заявок, вы можете сделать не более двух запросов.

Interaction Protocol

Взаимодействие начинает Ваша программа, задавая запрос в формате `?` id, где id — целое число, задающее номер заявки ($1 \le id < 10^9$). Программа жюри в ответ выдаст строку, в которой произведено форматное преобразование. Если вы хотите вывести ответ, выведите `!` s, где s — формат названия для данного отдела. Это действие запросом не считается.

Заметим, что в данной задаче ответ программы жюри может формироваться по мере поступления запросов (но при этом он всегда будет соответствовать всем ранее сделанным запросам), то есть интерактор **является адаптивным**.

Example

стандартный ввод	стандартный вывод
roga3333kopyta roga322kopyta	? 333 ? 22 ! roga3%02dkopyta