Задача A. 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.

Формат входных данных

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.

Формат выходных данных

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

Задача B. A + B

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

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

Здесь за $\&$, $\mid$ 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 $a + b$.

Формат входных данных

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

Формат выходных данных

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 $(a \,\&\, b \,\&\, x) + (a \mid b \mid x) + (a \oplus b \oplus x)$.

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».

Примеры

стандартный ввод	стандартный вывод
3	? 1 ! 2

Замечание

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

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.

Задача C. Программирование квадрокоптеров

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
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.

Формат входных данных

Это интерактивная задача.

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

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

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_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.

Формат выходных данных

Примеры

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> ! ((()))

Задача D. Интеракдивная задача

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

Это интерактивная задача.

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

Вы можете делать запросы следующего вида:

- `+ c`: эта команда осуществляет присвоение $x = x + c$ ($1 \leqslant c < n$), 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.

Протокол взаимодействия

Взаимодействие начинается с чтения целого числа n ($2 < n \leqslant 1000$), 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 $x = x + c$ ($1 \leqslant c < n$), 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.

Примеры

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

Замечание

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 $x = 5$, $\lfloor \frac{x}{n} \rfloor = 1$.

Задача E. 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.

Пример

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

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 F. Железнодорожная задача

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

Это — интерактивная задача

Места в плацкартных вагонах пронумерованы следующим образом: нечётные места являются нижними, чётные — верхними, места с номерами с 1 по 36 расположены в основной секции вагона, места с 37 по 54 являются боковыми. Точная нумерация мест приведена на следующей иллюстрации:

Программа жюри загадала некоторое место p.

Ваша задача — отгадать загаданное место с помощью запросов вида `?` d, где d — целое число от -54 до 54 включительно, которое надо прибавить к номеру текущего места. В начале взаимодействия номер текущего места совпадает с загаданным (то есть равен p).

Если номер места, полученный в результате сложения, оказывается меньше 1 или больше 54, то вы **сразу же** проигрываете. В противном случае программа жюри изменяет текущее место, прибавляя к нему d, и выдаёт информацию про это место: является ли оно верхним или нижним, а также является ли оно обычным или боковым.

Требуется не более чем за 6 запросов отгадать **исходное** место p.

Протокол взаимодействия

Взаимодействие начинает ваша программа, отправляя первый запрос.

Каждый запрос имеет формат `?` d, где $-54 \leqslant d \leqslant 54$ — число, которое вы хотите прибавить к номеру текущего места.

Если место некорректно, программа тут же завершается и решение получает вердикт Wrong Answer. В противном случае программа выводит строку из двух слов, разделённых пробелом. Первое слово — `low`, если место нижнее, и `high`, если место верхнее. Второе слово — `main`, если место расположено в основной секции вагона, и `side`, если место боковое.

Чтобы вывести ответ, используйте формат `!` P, где P — предполагаемое значение p. В случае, если ответ будет правильным ($P = p$), и число запросов не превышает 6, решение будет зачтено. Вывод ответа запросом не считается.

Гарантируется, что значение p определено в начале взаимодействия и не меняется в его процессе (то есть интерактор не является адаптивным).

Пример

стандартный ввод	стандартный вывод

`? -1`
 	
`? 5`
 	
`! 33`
	
	`high main`
	
	`low side`

Замечание

В примере из условия программа участника решила вывести ответ наугад, не имея абсолютной уверенности в значении p. В этом случае программе участника повезло, но всегда везти не будет...

Не забывайте после каждого запроса или вывода ответа выводить символ перевода строки, а также сбрасывать буфер ввода-вывода. Функция `print` в Python или вывод `endl` в C++ делают это автоматически. Можно также вызывать функцию `flush` используемого языка программирования.

Задача G. Ё322

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Это интерактивная задача

Для крупного телекоммуникационного оператора жизненно важным является вопрос диверсификации используемого оборудования с сохранением унификации программного обеспечения. Поэтому инженеры компании МТС решили исследовать вопрос о переносимости системного программного обеспечения между принципиально разными типами процессоров.

Планируется разработка модуля автоадаптации с кодовым наименованием «Ё322», который будет распознавать архитектуру микропроцессора и генерировать для неё эффективный код. Ваша задача в рамках этой разработки — написать модуль, который отличает два типа организации потоков ввода-вывода — стек и очередь.

Дана структура данных p, представляющая собой или стек, или очередь. Структура может быть пустой или содержать некоторое количество (не более 297) элементов.

Ваш модуль получает эксклюзивный доступ к структуре и может задать до 300 запросов одного из двух видов `put` x — добавить число x (целое число от 1 до 1000) в структуру данных или `get` — получить значение из головы очереди (вершины стека) и удалить соответствующий элемент. Если структура пуста, вы получаете специальное сообщение. После 300 запросов (или ранее) вы должны ответить, стек это или очередь.

Протокол взаимодействия

Взаимодействие начинает Ваша программа, выводя одну из команд. Если вы планируете поместить элемент x в структуру данных, выведите команду `put` x ($1 \leqslant x \leqslant 1000$). Если вы планируете снять элемент и узнать его значение, выведите команду `get`. Программа жюри выведет одно целое число — значение взятого элемента или -1, если структура пуста. Гарантируется, что изначально структура содержит не более 297 элементов.

Как только вы готовы определить, какая именно структура данных загадана, выведите `stack`, если это стек, и `queue`, если это очередь. Это действие не учитывается при подсчёте количества запросов.

Обратите внимание, что интерактор является адаптивным, то есть финальный ответ достраивается по ходу взаимодействия (но всегда удовлетворяет всей имеющейся информации и условиям задачи).

Пример

стандартный ввод	стандартный вывод
3	put 3 put 4 get queue

Задача H. Квадратный корень и квадратный квадрат

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Найдите такое число x, что $x^2 + \sqrt{x} = C$, с точностью не менее 6 знаков после точки.

Формат входных данных

В единственной строке содержится вещественное число $1.0 \leqslant C \leqslant 10^{10}$.

Формат выходных данных

Выведите одно число — искомый x.

Примеры

stdin	stdout
2.0000000000	1.000000000
18.0000000000	4.000000000

Задача I. Дремучий лес

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Чтобы помешать появлению СЭС в лагере, администрация ЛКШ перекопала единственную дорогу, соединяющую «Берендеевы поляны» с Судиславлем, теперь проехать по ней невозможно. Однако, трудности не остановили инспекцию, хотя для СЭС остаётся только одна возможность — дойти до лагеря пешком. Как известно, Судиславль находится в поле, а «Берендеевы поляны» — в лесу.

- Судиславль находится в точке с координатами $(0, 1)$.
- «Берендеевы поляны» находятся в точке с координатами $(1, 0)$.
- Граница между лесом и полем — горизонтальная прямая $y = a$, где a — некоторое число $(0 \leqslant a \leqslant 1)$.
- Скорость передвижения СЭС по полю составляет V_p, скорость передвижения по лесу — V_f. Вдоль границы можно двигаться как по лесу, так и по полю.

Администрация ЛКШ хочет узнать, сколько времени у неё осталось для подготовки к визиту СЭС. Она попросила вас выяснить, в какой точке инспекция СЭС должна войти в лес, чтобы дойти до «Берендеевых полян» как можно быстрее.

Формат входных данных

В первой строке входного файла содержатся два положительных целых числа — V_p и V_f $(1 \leqslant V_p, V_f \leqslant 10^5)$. Во второй строке содержится единственное вещественное число — координата по оси Oy границы между лесом и полем a $(0 \leqslant a \leqslant 1)$

Формат выходных данных

В единственной строке выходного файла выведите вещественное число с точностью не менее 7 знаков после запятой — координата по оси Ox точки, в которой инспекция СЭС должна войти в лес.

Примеры

stdin	stdout
5 3 0.4	0.783310604
5 5 0.5	0.500000000