

Problem A. А какой у меня дивизион?

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Отборочно-распределительный этап Рукода обычно имеет следующую структуру. Всего есть четыре дивизиона интенсивов, расположенные в порядке возрастания сложности как F, E, D, C.

- Первые p_f задач — базовые задачи, проверяющие владение основными конструкциями языка (темы дивизиона F).
- Далее идут p_e задач из тем дивизиона E, по одной задаче на тему.
- Далее идут p_d задач из тем дивизиона D, по одной задаче на тему.
- Далее идёт одна задача из одной из тем дивизиона C.

Участники, не решившие ни одной задачи, могут участвовать только в дивизионе F.

Участникам, решившим ровно одну задачу, рекомендуется участие в дивизионе E.

Участникам, решившим все задачи, рекомендуется участие в Чемпионате в дивизионе AB.

Для остальных рекомендации строятся так: если количество решённых участником задач таково, что он **гарантированно** решил более $\lfloor (p_d + 1)/2 \rfloor$ задач дивизиона D (или **более сложных**), ему рекомендуется участие в дивизионе C. Если он решил ровно $\lfloor (p_d + 1)/2 \rfloor$ задачи, ему рекомендуется самому выбрать между дивизионами C и D, то есть рекомендация выглядит как C/D. В противном случае применяется аналогичное рассуждение с заменой дивизиона D дивизионом E, а дивизиона C дивизионом D.

Вам даются числа p_f , p_e и p_d , а также количество решённых участником задач. Выведите рекомендации для участников по выбору дивизиона интенсивов.

Input

Первая строка входных данных содержит одно целое число p_f ($1 \leq p_f \leq 4$ — количество задач на темы дивизиона f). Вторая строка входных данных содержит одно целое число p_e ($3 \leq p_e \leq 7$) — количество задач на темы дивизиона e . Третья строка входных данных содержит одно целое число p_d ($3 \leq p_d \leq 7$) — количество задач на темы дивизиона d . Четвёртая строка входных данных содержит одно целое число k — количество задач, решённых участником ($0 \leq k \leq p_d + p_e + p_f + 1$).

Output

Если участник может участвовать только в дивизионе F, выведите «F». Если участнику рекомендуется дивизион E, выведите «E». Если участнику рекомендуется самому выбирать между E и D, выведите «D/E». Если участнику рекомендуется дивизион D, выведите «D». Если участнику рекомендуется самому выбирать между D и C, выведите «C/D». Если участнику рекомендуется дивизион C, выведите «C». Если участнику рекомендуется участвовать сразу в Чемпионате в дивизионе AB, выведите «Champ AB».

Examples

<i>standard input</i>	<i>standard output</i>
2 5 5 0	F
2 5 5 4	E
2 5 5 5	D/E
2 5 5 9	D
2 5 5 10	C/D
2 5 5 12	C

Problem B. Большие рамки

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Рассмотрим шарнирные конструкции из четырёх планок, соединённых в виде четырёхугольника, обладающие следующим свойством:

- Длины планок a , b , c и d — целые числа.
- $1 \leq d < c < b < a$.
- a , b , c и d являются элементами последовательности Фибоначчи ($F_1 = 1$, $F_2 = 2$, $F_i = F_{i-1} + F_{i-2}$ для $i < 2$).
- Планки соединены в четырёхугольник (концы a и c соединены с концами b и d шарнирами так, что при обходе против часовой стрелки получается a , b , c , d), при этом существует положение планок, при котором площадь четырёхугольника ненулевая.

Назовём такие конструкции *рамками Фибоначчи*.

Упорядочим все рамки Фибоначчи по возрастанию периметра. Ваша задача — найти остаток от деления периметра N -й рамки Фибоначчи на 998 244 353.

Input

Первая строка входных данных содержит одно целое число N ($1 \leq N \leq 10^9$).

Output

Выведите одно целое число — остаток от деления периметра N -й рамки Фибоначчи на 998 244 353.

Examples

<i>standard input</i>	<i>standard output</i>
1	11
2023	721775497

Problem C. Валидация валидатора

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Ваши коллеги придумали интересную задачу и хотят предложить её на очередной чемпионат RuCode. Вам, как самому опытному в компании специалисту по спортивному программированию, предложено проанализировать написанный валидатор, после чего в соответствии с валидатором проверить тесты на корректность и полноту.

Тесты к задаче состоят из L целых чисел, расположенных по одному на строку.

Для каждой из L переменных в валидаторе заданы ограничения. Вам даются строки кода из валидатора вида `inf.readInt (min, max, "N");`, обозначающие, что переменная с именем N может принимать целые значения от min до max включительно. i -й сверху строке валидатора соответствует i -е число в каждом тесте.

Далее вам даются тесты. Ваша задача — проверить, что все переменные в тестах удовлетворяют заданным валидатором ограничениям, а также проверить, что для каждой переменной существует тест, в котором она принимает максимальное значение, и тест, в котором она принимает минимальное значение.

Input

Первая строка входных данных содержит одно целое число L ($1 \leq L \leq 10$) — количество переменных в одном тесте к проверяемой задаче.

i -я из последующих L строк содержит одну строку валидатора, задающую диапазон i -й переменной и имеющую формат `inf.readInt (l , r , "name");`, где l_i — нижняя граница переменной, r_i — верхняя граница переменной, $name$ — имя переменной ($1 \leq l \leq r \leq 10^6$, $name$ состоит из одной или двух строчных или заглавных латинских букв). Гарантируется, что все пробелы (после `readInt` и после обеих запятых внутри скобок) являются одиночными.

Следующая строка содержит одно целое число T ($1 \leq T \leq 500$) — количество тестов в проверяемой задаче. i -я из последующих T строк описывает тест с номером i и содержит по L целых чисел $a_{i,j}$ ($1 \leq a_{i,j} \leq 10^6$). j -е из этих чисел соответствует переменной из j -го вызова функции `inf.readInt()`.

Output

Если какой-то из проверяемых тестов некорректен (то есть i -я переменная меньше l_i или больше r_i), выведите в первой строке сообщение **Error**, во второй строке номер первого некорректного теста (начиная с 1), и затем имя первой встретившейся в этом тесте некорректной переменной.

Если все тесты корректны, но ни в одном тесте набора не достигается минимум хотя бы по одной переменной, выведите в первой строке сообщение **Warning: Min**, затем через одиночный пробел — имена переменных (в порядке описания в валидаторе), для которых минимум не достигается. Если все минимумы достигаются, выведите одну строку **Min OK**.

Если все тесты корректны, но ни в одном тесте набора не достигается максимум хотя бы по одной переменной, выведите во второй строке сообщение **Warning: Max**, затем через одиночный пробел — имена переменных (в порядке описания в валидаторе), для которых максимум не достигается, перечисленные через одиночный пробел. Если все максимумы достигаются, выведите одну строку **Max OK**.

В случае проблем с точным пониманием формата вывода смотрите примеры к задаче.

Examples

<i>standard input</i>	<i>standard output</i>
<pre>2 inf.readInt (3, 14, "P"); inf.readInt (15, 26, "i"); 3 3 17 5 26 13 20</pre>	<pre>Warning: Min i Warning: Max P</pre>
<pre>1 inf.readInt (1, 10, "N"); 1 1000000</pre>	<pre>Error 1 N</pre>
<pre>2 inf.readInt (3, 14, "P"); inf.readInt (15, 26, "i"); 3 3 17 5 26 15 1</pre>	<pre>Error 3 P</pre>
<pre>2 inf.readInt (3, 14, "P"); inf.readInt (15, 26, "i"); 5 3 17 5 26 14 20 5 26 11 15</pre>	<pre>Min OK Max OK</pre>

Problem D. Грамотные разбиения

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

На вещественной прямой расположено $2n$ различных точек, пронумерованных от 1 до $2n$. Будем говорить, что способ разбить их на n пар $(a_1, b_1), \dots, (a_n, b_n)$ *грамотный*, если точки во всех парах можно соединить кривыми так, что все n кривых не пересекаются друг с другом и не пересекают вещественную прямую. Сколько существует грамотных способов разбить точки на пары? Поскольку это число может быть большим, выведите его по модулю p — заданного простого числа.

Два способа разбиения считаются одинаковыми, если можно переставить пары и переставить точки в парах так, чтобы из одного способа получить другой.

Input

В единственной строке записаны два целых числа N и p : максимальное число пар точек и простой модуль ($1 \leq N \leq 200$; $10^8 < p < 10^9$).

Output

Выведите N строк: ответ на задачу для $n = 1, 2, \dots, N$. Каждый ответ выведите по модулю p .

Example

<i>standard input</i>	<i>standard output</i>
5 998244353	1 3 14 84 592

Problem E. Доска и фигуры

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

У Алисы и Боба есть набор, состоящий из обычных и сказочных шахматных фигур. Всего в наборе 6 типов фигур (слон, ладья, ферзь, архиепископ, канцлер и магараджа). В наборе ровно две фигуры каждого типа. Ладья может перемещаться на любое количество клеток по горизонтали или вертикали. Слон может перемещаться на любое количество клеток по диагонали. Ферзь объединяет силу ладьи и слона и может перемещаться на любое количество клеток по горизонтали, вертикали или диагонали. Конь перемещается на любую из ближайших клеток, которые не находятся на той же горизонтали, вертикали или диагонали. Таким образом, ход образует форму буквы «Г»: две клетки по вертикали и одна клетка по горизонтали, или две клетки по горизонтали и одна клетка по вертикали. Архиепископ ходит как слон и конь, канцлер как ладья и конь, а магараджа как ферзь и конь.

Алиса и Боб решили сыграть в игру на шахматной доске 8 на 8. В начале игры фигуры перемещаются и выставляются в ряд, определяя порядок. Первую фигуру в заданном порядке на доску выставляет Алиса, вторую Боб, третью Алиса, и так далее. Фигуру нужно выставить на свободное поле так, чтобы она не била и не была под боем выставленных ранее фигур. Игрок, который не может сделать ход, проигрывает.

Ваша задача — написать программу, вычисляющую, кто победит при оптимальной игре обоих противников.

Input

Первая строка содержит 12 заглавных английских букв, задающих порядок выставления фигур на доску. «B» соответствует слону, «R» — ладье, «Q» — ферзю, «A» — архиепископу, «C» — канцлеру, «M» — магарадже. В заданном наборе ровно две фигуры каждого из шести типов.

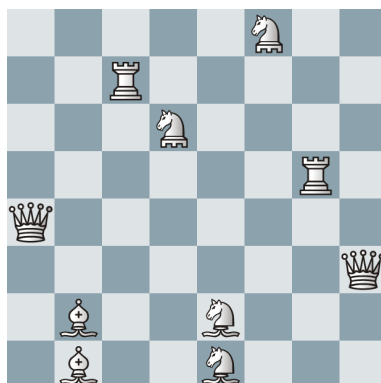
Output

Выведите «Alice», если побеждает Алиса. Выведите «Bob», если побеждает Боб.

Examples

<i>standard input</i>	<i>standard output</i>
BBAARRCCQMM	Bob
BAMBAMQQRCCR	Alice

Note



Возможная конечная позиция для расстановки из первого примера, если соперники играли не оптимально. Первые десять фигур (все, кроме магарадж) выставлены на доску. На d6 и f8 стоят канцлеры. На e1 и e2 стоят архиепископы. Одиннадцатую фигуру (магараджу) поставить некуда.

Problem F. Естественные числа

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Смешанная позиционная система счисления задаётся последовательностью оснований b_i , где b_i — целое число, не меньшее 2.

Чтобы получить представление числа X в такой системе счисления, применяется следующий алгоритм: $x_0 = x$, $a_i = x_i \bmod b_{i+1}$, $x_{i+1} = \lfloor x_i / b_{i+1} \rfloor$.

Иначе говоря, числа в такой системе представляются как

$$a_0 + b_1 \cdot (a_1 + b_2 \cdot (a_2 + \dots + b_{n-1} \cdot a_{n-1}) \dots)$$

где $0 \leq a_i < b_{i+1}$ для всех i от 0 до $n - 1$.

Таким образом, максимальное число, которое можно представить в системе, заданной последовательностью b_i , равно $\prod_{i=1}^n b_i - 1$.

Назовём целое положительное число k *естественным*, если верно следующее правило: число в системе счисления, заданной последовательностью b_i , делится на k тогда и только тогда, когда сумма цифр этого числа (то есть сумма всех a_i) также делится на k (аналог делимости на 3 и на 9 в десятичной системе счисления). В частности, так как все числа делятся на 1, то единица будет естественным числом для любой последовательности оснований.

По заданной последовательности оснований b_i найдите все естественные числа для соответствующей смешанной системы счисления.

Input

Первая строка входных данных содержит одно целое число N — длину последовательности оснований ($2 \leq N \leq 10^5$).

Вторая строка содержит N целых чисел b_i ($2 \leq b_i \leq 10^9$) — элементы последовательности оснований, перечисленные в том же порядке, в котором они идут в последовательности.

Output

Выведите все естественные числа для данной последовательности, отсортированные по возрастанию, по одному числу на строку.

Examples

<i>standard input</i>	<i>standard output</i>
3 10 10 10	1 3 9
4 15 10 20 23	1

Problem G. Ёлочные игрушки

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

У вас есть ёлочные игрушки: шарики, покрашенные в n различных цветов. Для каждого цвета i от 1 до n есть ровно x_i шариков этого цвета. Вы играете в игру, состоящую из действий. За одно действие вы можете взять k шариков попарно различных цветов и выкинуть их. Какое наибольшее количество действий вы можете совершить?

Input

В первой строке записаны два целых числа n и k : число цветов и количество шариков, выкидываемых при каждом действии ($1 \leq k \leq n \leq 2 \cdot 10^5$). Во второй строке содержатся n чисел x_i , разделённых пробелами — количество шариков i -го цвета ($1 \leq x_i \leq 10^9$).

Output

Выведите строку, в которой будет одно целое число — наибольшее возможное количество действий.

Examples

<i>standard input</i>	<i>standard output</i>
4 3 5 8 9 4	8
10 5 1 2 3 4 5 6 239 239 239 239	21

Problem H. Жаркий день

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Одним знойным летним днём Татьяна с ребятами пешком отправилась на озеро. Дорога была столь утомительна, что все обрадовались, когда за ними после купания внезапно прилетела на вертолёт их подруга Ольга.

В вертолёт доступны два ряда с пассажирскими сиденьями. Для уменьшения количества перелётов Оля разрешила взрослому к себе на колени посадить ребёнка на всех пассажирских сиденьях, кроме центрального заднего. А по центру заднего ряда сможет сесть либо один взрослый, либо один ребёнок.

Ниже представлена схема, на которой указано, на каких местах разрешено сидеть взрослому с ребёнком (максимально 2 пассажира на одном кресле), а где может располагаться максимум один человек: взрослый или ребёнок. А также отмечено место пилота — Ольги.

О.		max 2	
max 2	max 1	max 2	

Какое минимальное количество перелётов должна совершить Ольга, чтобы всех вернуть домой? Перемещение в одну сторону считается за отдельный перелёт.

Input

В первой строке через пробел записаны целые числа n и m — количество взрослых и детей, которые отправились на озеро пешком ($1 \leq n, m \leq 10^6$).

Output

Выведите минимальное количество перелётов для возвращения всех домой.

Examples

<i>standard input</i>	<i>standard output</i>
1 1	1
1 4	1
5 5	3

Problem I. Запрещённое множество

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

Задано множество десятичных цифр. Найдите минимальное простое число, обладающее следующим свойством: в десятичной записи этого числа **нет** цифр из данного множества.

Например, если задано множество $\{0, 6, 3, 9\}$, то простое число 71 удовлетворяет требованию задачи (кроме, быть может, минимальности), а простое число 101 — нет (есть цифра 0, которая присутствует во множестве).

Input

Первая строка содержит одно целое число n — количество цифр во множестве ($1 \leq n \leq 10$). Каждая из последующих n строк содержит одно целое число d_i ($0 \leq d_i \leq 9$) — очередной элемент множества. Гарантируется, что все d_i попарно различны.

Output

Если не существует простых чисел, в десятичной записи которых отсутствуют цифры из заданного множества, выведите -1 . Иначе выведите наименьшее такое число.

Examples

standard input	standard output
7 0 1 2 4 6 8 9	3
9 0 1 2 3 5 6 7 8 9	-1

Problem J. Испытания и сбои

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Для ветроустановки, расположенной в районах Крайнего Севера, изготовлено устройство, автоматически считывающее показания датчиков.

В настоящий момент проходят испытания устройства. При тестовом запуске устройство в случайный момент времени считывает время со стоящего перед ним шестиразрядного таймера (равновероятны все значения от 00:00:00 до 23:59:59). Разряды таймера занумерованы слева направо, начиная с единицы. Против каждого разряда расположен считывающий элемент, «отвечающий» за этот разряд.

Известно, что ровно один считывающий элемент устройства сбоит. Если сбойный элемент отвечает за цифру, которая находится с краю (первую или шестую), то при каждом считывании с равной вероятностью ($1/2$) или время передаётся без искажений, или сбойная цифра меняется местами с соседней (то есть, например, время 12:34:32 при сбойном шестом считывающем элементе прочитается или как 12:34:32, или как 12:34:23).

Если сбойный элемент отвечает за цифру в другой позиции (с 2 по 5 включительно), то при каждом считывании с равной вероятностью ($1/3$) передаётся или правильное время, или сбойная цифра меняется с предыдущей, или сбойная цифра меняется с последующей.

Например, если сбойный элемент находится в позиции 2, то с равной вероятностью 12:34:32 будет передано или как 12:34:32, или как 21:34:32, или как 13:24:32.

После этого считанное время проверяется на корректность. Если считанное время является недопустимым (то есть количество часов больше 23 или количество минут или секунд больше 59), или же такое время уже было передано, то данные **не** передаются до следующего считывания.

Тестовый запуск идёт до тех пор, пока не будет передано 3600 результатов.

По считанным результатам определите, какой считывающий элемент устройства сбоит.

Input

Первая строка входных данных содержит одно целое число 3600.

Каждая из последующих строк содержит шесть цифр — переданную информацию в порядке её передачи. Двухзначное число, сформированное первыми двумя цифрами, не превосходит 23, двухзначное число, сформированное третьей и четвёртой цифрами, не превосходит 59, двухзначное число, сформированное пятой и шестой цифрами, также не превосходит 59.

Гарантируется, что данные сгенерированы описанным в условии способом (случайный момент времени на таймере, возможные искажения, повторная генерация в случае дубля или некорректных дат) и впоследствии отсортированы лексикографически.

Output

Выведите одно число от 1 до 6 — номер разряда (считая слева направо), которому соответствует сбойный считывающий элемент.

Example

<i>standard input</i>	<i>standard output</i>
3600 000000 000034 000046 ... 235744 235833 235845	2

Note

Полный вариант примера можно найти в аттачментах к задаче.

Problem K. йыньливарпеН порядок

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

Это задача с двойным запуском.

У Вас есть массив из $n \leq 1000$ беззнаковых 64-битных целых чисел. Вы собираетесь быстро переслать его на другой компьютер. Для этого вы параллельно отправляете каждое число через Интернет и потом снова собираете массив.

Однако возникла непредвиденная проблема. Как известно, «сетевой» порядок байт в многобайтовом числе отличается от используемого в современных компьютерах: если в современном компьютере байты записываются от младшего к старшему (little-endian), то при сетевом порядке байты записываются от старшего к младшему (big-endian). При преобразовании каждое число записывается как последовательность из 8 байт, и байты записываются в обратном порядке. И в некоторых случаях из-за сбоя на серверах обратное преобразование сделано не было...

То есть каждый элемент массива может прийти либо в обычном little-endian порядке, либо в сетевом big-endian. Порядок элементов массива сохранён. При передаче вы можете использовать не более 1024 64-битных чисел (то есть не более 8 **кибибайт**). Ваша задача — передать исходный массив и восстановить его после возможных искажений.

Input

Если требуется передать массив, то в первой строке записано слово «**encode**», во второй — целое число n ($1 \leq n \leq 1000$), а в третьей — n целых чисел в интервале от 0 до $2^{64} - 1$.

Если требуется получить массив, то в первой строке записано слово «**decode**», во второй — целое число k ($k \leq 1024$), а в третьей — k целых чисел в интервале от 0 до $2^{64} - 1$. Гарантируется, что числа идут в том же порядке, в котором они были переданы, и что любое из чисел либо передаётся неизменным, либо имеет перевёрнутый порядок байтов (в соответствии с условием задачи).

Output

В случае передачи массива в первой строке выведите одно целое число $k \leq 1024$: количество передаваемых чисел. Во второй строке выведите k целых чисел в интервале от 0 до $2^{64} - 1$.

В случае получения массива выведите n целых чисел — исходный массив.

Examples

standard input	standard output
encode 3 15 10 2023	6 15 15 10 10 2023 2023
decode 6 15 15 10 720575940379279360 16647274547598327808 2023	15 10 2023

Note

В нижнем примере — в случае получения массива — все шесть чисел будут даны в одной строке. Дополнительный перевод строки добавлен только для удобства чтения.

На каждом тесте ваша программа будет запущена два раза: в первый раз на передачу массива, а во второй раз на получение, причём в качестве входных данных будет передан вывод первого запуска.

ка с возможными искажениями. Тест считается пройденным, если исходный массив восстановлен корректно.

Problem L. Клетчатое поле

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Рассмотрим бесконечное во все стороны клетчатое поле. Каждая клетка покрашена либо в чёрный цвет, либо в белый. У каждой чёрной клетки **ровно три** соседние по стороне клетки — также чёрные.

Будем рассматривать периодические раскраски. А именно, сначала раскрасим какой-то прямоугольник из клеток. Затем разобьём поле на такие прямоугольники, стыкуя их по сторонам. Все прямоугольники раскрасим одинаково.

Приведите пример раскраски, в которой доля чёрных клеток равна заданному рациональному числу p/q , или выясните, что это невозможно.

Input

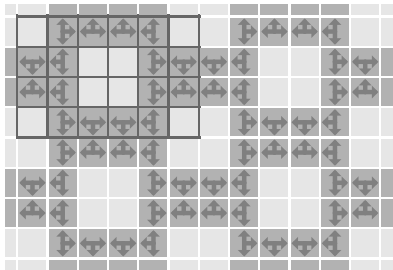
В первой строке заданы два целых числа p и q — числитель и знаменатель требуемой доли чёрных клеток ($0 \leq p \leq q \leq 10$; числа p и q взаимно просты).

Output

Если раскраска возможна, в первой строке выведите два целых числа h и w : высоту и ширину прямоугольника ($1 \leq h, w \leq 1000$). Далее выведите раскраску прямоугольника из h строк, по w символов в каждой. Символ «.» (точка) обозначает белую клетку, символ «#» (решётка) — чёрную. Отношение количества чёрных клеток в прямоугольнике к общему количеству клеток в нём должно быть равно p/q . Если возможных раскрасок несколько, выведите любую из них.

Если раскраска невозможна, в первой строке выведите «-1 -1».

Examples

<i>standard input</i>	<i>standard output</i>	<i>illustration</i>
2 3	4 6 .####. ##.## ##.## .####.	
1 1	-1 -1	

Problem M. Лишь одно число

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

На доске написаны целые числа от ℓ до r включительно, каждое — ровно один раз. За один шаг можно выбрать на доске два числа a и b , полусумма $\frac{a+b}{2}$ которых — **целое** число, стереть с доски выбранные два числа и написать на доске их полусумму.

Может ли после нескольких (нуля или более) шагов на доске остаться ровно одно число? Если да, то какое максимальное число может при этом получиться?

Input

В первой строке заданы два целых числа ℓ и r — минимальное и максимальное числа, изначально записанные на доске ($1 \leq \ell \leq r \leq 100$).

Output

Выведите максимальное число, которое может в одиночестве остаться на доске. Если на доске не может получиться одно число, выведите -1 .

Example

<i>standard input</i>	<i>standard output</i>	<i>explanation</i>
2 4	3	<u>2</u> , 3, <u>4</u> \rightarrow <u>3</u> , 3 \rightarrow 3

Problem N. Множества и диапазоны

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

Это интерактивная задача.

У вас есть n множеств целых чисел. Множества пронумерованы от 1 до n . Изначально все множества пустые. Ваша задача — выполнить q запросов следующих трёх типов.

1. «+ ℓ r x ». Добавить x во все множества с номерами от ℓ до r включительно ($1 \leq \ell \leq r \leq n$).
2. «- ℓ r x ». Удалить x из всех множеств с номерами от ℓ до r включительно ($1 \leq \ell \leq r \leq n$).
3. «? k ». Вывести размер множества с номером k ($1 \leq k \leq n$).

Множества ведут себя как обычные множества: если мы добавляем элемент, который уже есть, или удаляем элемент, которого уже нет, ничего не происходит.

Input

Первая строка содержит два целых числа n и q : количество наборов и количество запросов соответственно ($1 \leq n \leq 10^9$; $0 \leq q \leq 10^5$). Следующие q строк содержат запросы в описанном выше формате ($1 \leq x \leq q$ в запросах).

Output

Для каждого запроса третьего типа ваша программа должна вывести ответ на отдельной строке. Гарантируется, что количество таких запросов не превосходит 10 000. После вывода каждого ответа не забывайте выводить перевод строки и очищать буфер вывода, иначе решение получит вердикт «Idleness Limit Exceeded». Очистить буфер можно, например, вызовом `fflush (stdout)` в C или `C++.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python.

Example

<i>standard input</i>	<i>standard output</i>
736 10	0
? 1	1
+ 1 5 1	2
+ 2 600 2	2
? 1	2
? 2	1
+ 1 6 2	
? 1	
? 2	
- 1 6 2	
? 4	