Задача A. Целочисленное деление

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Заданы два целых числа A и B. Результатом целочисленного деления A на B будем считать целое число Q такое, что $A = Q \cdot B + R$ и при этом $0 \leqslant R < |B|$. Выведите Q.

Формат входных данных

В первой строке ввода заданы два целых числа A и B ($-2^{31} \leqslant A, B < 2^{31}$).

Формат выходных данных

Выведите одно число — результат целочисленного деления A на B. Если $B = 0$, выведите вместо этого строку «`Na nol' delit' nel'zya!!!`».

Примеры

стандартный ввод	*стандартный вывод*
`2 3`	`0`
`239 0`	`Na nol' delit' nel'zya!!!`

Задача B. Целочисленное деление-64

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Заданы два целых числа A и B. Результатом целочисленного деления A на B будем считать целое число Q такое, что $A = Q \cdot B + R$ и при этом $0 \leqslant R < |B|$. Выведите Q.

Формат входных данных

В первой строке ввода заданы два целых числа A и B ($-2^{63} \leqslant A, B < 2^{63}$).

Формат выходных данных

Выведите одно число — результат целочисленного деления A на B. Если $B = 0$, выведите вместо этого строку «`Na nol' delit' nel'zya!!!`».

Примеры

стандартный ввод	*стандартный вывод*
`2 3`	`0`
`239 0`	`Na nol' delit' nel'zya!!!`

Задача C. Умножение и деление на 2

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Рассмотрим число x, хранящееся в типе данных `uint32`. Разрешается умножать и делить его на 2 в любом порядке сколько угодно раз. Можно ли этими операциями получить число y?

При умножении на 2 число a в типе данных `uint32` заменяется на $(a \cdot 2) \bmod 2^{32}$. Например, $(3 \cdot 2) \bmod 2^{32} = 6$, а $(2\,147\,483\,649 \cdot 2) \bmod 2^{32} = 2$.

При делении на 2 число a в типе данных `uint32` заменяется на $\left\lfloor \frac{a}{2} \right\rfloor$. Например, $\left\lfloor \frac{6}{2} \right\rfloor = 3$, а $\left\lfloor \frac{3}{2} \right\rfloor = 1$.

Формат входных данных

В первой строке задано целое число t — количество тестовых случаев ($1 \leqslant t \leqslant 1000$). В следующих t строках заданы тестовые случаи, по одному в строке. Каждый тестовый случай задаётся двумя целыми числами x и y ($0 \leqslant x, y < 2^{32}$).

Формат выходных данных

В ответ на каждый тестовый случай выведите одно слово в отдельной строке: «`Yes`», если из x можно указанными операциями получить y, и «`No`» в противном случае.

Пример

стандартный ввод	*стандартный вывод*
2 2147483649 1 9 13	Yes No

Пояснение к примеру

В первом тестовом случае мы можем умножить x на 2, а результат поделить на 2 и получить y.

Во втором тестовом случае не существует способа превратить x в y.

Задача D. Магия вуду

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Чтобы наложить на пирамиду заклятие, шаман Эхнатона должен использовать один или несколько магических кристаллов. Каждый кристалл характеризуется целым числом — своей силой. От силы используемых кристаллов зависит мощность заклятия: при использовании k кристаллов, имеющих силы f_1, f_2, …, f_k, заклятие будет иметь силу $(f_1 \cdot f_2 \cdot \ldots \cdot f_k) \bmod m$, где m — магическое число вуду, равное $2^{30} = 1\,073\,741\,824$. Выражение «$a \bmod b$» означает остаток от деления a на b.

Шаман уже убедился в ваших способностях к вычислениям, и дал новое задание. Он снабдил вас информацией об имеющихся у него магических кристаллах. Теперь шаман просит сказать ему, какой максимальной мощности может достичь заклятие при правильном выборе из имеющихся кристаллов одного или нескольких, которые надо использовать.

Формат входных данных

В первой строке записано целое число n — количество магических кристаллов, имеющихся в распоряжении шамана ($1 \leqslant n \leqslant 12$). Во второй строке записаны n целых чисел p_1, p_2, …, p_n через пробел — силы магических кристаллов ($1 \leqslant p_i \leqslant 10\,000$).

Формат выходных данных

В первой строке выведите одно число — максимальную возможную силу заклятия.

Примеры

стандартный ввод	*стандартный вывод*
2 1 3	3
3 10000 11 9999	99990000

Задача E. Сложение и переворачивание

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Рассмотрим целое неотрицательное число x, которое хранится в 32 битах памяти:

$$x = b_{31} \cdot 2^{31} + b_{30} \cdot 2^{30} + \ldots + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0,$$

где каждый бит b_i может независимо от остальных принимать два значения: 0 и 1.

Произведём над числом последовательность операций, возможно, пустую. За одну операцию разрешается либо прибавить к числу единицу, либо развернуть составляющие его биты: 31-й бит поменять местами с нулевым, 30-й с первым, …, 16-й с 15-м. Можно выполнить любое количество любых операций в любом порядке.

За какое минимальное количество операций можно из нуля получить заданное число n?

Сложение производится по модулю 2^{32}, то есть, если текущее число равно $2^{32} - 1$, то после прибавления единицы число станет нулём.

Формат входных данных

В единственной строке задано целое число n ($0 \leqslant n < 2^{32}$).

Формат выходных данных

Выведите одно число: минимальное количество операций, которое потребуется, чтобы получить из нуля заданное число n.

Примеры

стандартный ввод	*стандартный вывод*
5	5
2147483648	2

Пояснения к примерам

В первом примере самый быстрый способ получить число 5 — это пять раз прибавить единицу.

Во втором примере сначала получим из нуля единицу, а затем развернём составляющие число биты, превратив $1 = 2^0$ в $2\,147\,483\,648 = 2^{31}$.

Задача F. йыньливарпеН порядок

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Это задача с двойным запуском.

У Вас есть массив из $n \leqslant 1000$ 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.

Формат входных данных

Если требуется передать массив, то в первой строке записано слово «`encode`», во второй — целое число n ($1 \leqslant n \leqslant 1000$), 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 ($k \leqslant 1024$), 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).

Формат выходных данных

В случае передачи массива в первой строке выведите одно целое число $k \leqslant 1024$: 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.

Примеры

стандартный ввод	*стандартный вывод*
encode 3 15 10 2023	6 15 15 10 10 2023 2023
decode 6 15 15 10 720575940379279360 16647274547598327808 2023	15 10 2023

Замечание

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

Задача G. Представление 16-битного целого числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано целое число. Выведите представление этого числа в 16-битном двоичном дополнительном коде.

Формат входных данных

В первой строке задано одно целое число n в десятичной записи ($-2^{15} \leqslant n < 2^{15}$).

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись дополнительного кода для числа n. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Примеры

стандартный ввод	*стандартный вывод*
3	00000000 00000011 00 03
-57	11111111 11000111 FF C7

Задача H. Представление 32-битного целого числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано целое число. Выведите представление этого числа в 32-битном двоичном дополнительном коде.

Формат входных данных

В первой строке задано одно целое число n в десятичной записи ($-2^{31} \leqslant n < 2^{31}$).

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись дополнительного кода для числа n. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Примеры

стандартный ввод	*стандартный вывод*
3	00000000 00000000 00000000 00000011 00 00 00 03
-57	11111111 11111111 11111111 11000111 FF FF FF C7

Задача I. Представление 64-битного целого числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано целое число. Выведите представление этого числа в 64-битном двоичном дополнительном коде.

Формат входных данных

В первой строке задано одно целое число n в десятичной записи ($-2^{63} \leqslant n < 2^{63}$).

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись дополнительного кода для числа n. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Примеры

стандартный ввод
3
стандартный вывод
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000011 00 00 00 00 00 00 00 03

стандартный ввод
-57
стандартный вывод
11111111 11111111 11111111 11111111 11111111 11111111 11111111 11000111 FF FF FF FF FF FF FF C7

Задача J. Представление 32-битного вещественного числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано вещественное число. Выведите представление этого числа в 32-битном типе `float`. В этом типе 1 старший бит хранит знак числа, следующие 8 битов отведены под экспоненту, а в оставшихся 23 битах находится мантисса.

Формат входных данных

В первой строке заданы два целых числа p и q через пробел ($|p|, |q| \leqslant 10^4$, $q \neq 0$). Число, которое нужно представить в типе `float` — это $\frac{p}{q}$. Помните, что если результат деления нельзя сохранить точно, в результирующей переменной типа `float` должно оказаться наиболее близкое из тех чисел, которые в этом формате можно хранить.

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись представления данного числа. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Число 0 следует записывать с положительным (то есть нулевым) знаковым битом.

Примеры

стандартный ввод	*стандартный вывод*
9 2	01000000 10010000 00000000 00000000 40 90 00 00
5 -3	10111111 11010101 01010101 01010101 BF D5 55 55

Задача K. Представление 64-битного вещественного числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано вещественное число. Выведите представление этого числа в 64-битном типе `double`. В этом типе 1 старший бит хранит знак числа, следующие 11 битов отведены под экспоненту, а в оставшихся 52 битах находится мантисса.

Формат входных данных

В первой строке заданы два целых числа p и q через пробел ($|p|, |q| \leqslant 10^9$, $q \neq 0$). Число, которое нужно представить в типе `double` — это $\frac{p}{q}$. Помните, что если результат деления нельзя сохранить точно, в результирующей переменной типа `double` должно оказаться наиболее близкое из тех чисел, которые в этом формате можно хранить.

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись представления данного числа. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Число 0 следует записывать с положительным (то есть нулевым) знаковым битом.

Примеры

стандартный ввод
`9 2`
стандартный вывод
`01000000 00010010 00000000 00000000 00000000 00000000 00000000 00000000` `40 12 00 00 00 00 00 00`

стандартный ввод
`5 -3`
стандартный вывод
`10111111 11111010 10101010 10101010 10101010 10101010 10101010 10101011` `BF FA AA AA AA AA AA AB`

Задача L. Представление 80-битного вещественного числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Задано вещественное число. Выведите представление этого числа в 80-битном типе `long double` (например, такой тип есть в компиляторе GNU C++). В этом типе 1 старший бит хранит знак числа, следующие 15 битов отведены под экспоненту, а в оставшихся 64 битах находится мантисса.

Формат входных данных

В первой строке заданы два целых числа p и q через пробел ($|p|, |q| \leqslant 10^9$, $q \neq 0$). Число, которое нужно представить в типе `long double` — это $\frac{p}{q}$. Помните, что если результат деления нельзя сохранить точно, в результирующей переменной типа `long double` должно оказаться наиболее близкое из тех чисел, которые в этом формате можно хранить.

Формат выходных данных

Выведите две строки. В первой строке выведите двоичную запись представления данного числа. Во второй строке выведите шестнадцатеричную запись. Отделяйте записи соседних байтов пробелами. Следуйте формату, указанному в примере.

Число 0 следует записывать с положительным (то есть нулевым) знаковым битом.

Примеры

стандартный ввод
`9 2`
стандартный вывод
`01000000 00000001 10010000 00000000 00000000 00000000 00000000 00000000 00000000 00000000` `40 01 90 00 00 00 00 00 00 00`

стандартный ввод
`5 -3`
стандартный вывод
`10111111 11111111 11010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101` `BF FF D5 55 55 55 55 55 55 55`

Задача M. Разность между значениями

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

В выражении

$$f(x) = ax + \frac{x}{b}$$

известны коэффициенты a и b. Найдите разность значений функции $f(x)$ в двух точках x_1 и x_2 при условии, что $|x_2 - x_1| = 1$.

Формат входных данных

В первой строке заданы через пробел четыре целых числа a, b, x_1 и x_2 ($1 \leqslant a, b, x_1, x_2 \leqslant 10^9$). Гарантируется, что числа x_1 и x_2 отличаются ровно на единицу.

Формат выходных данных

Выведите одно число — требуемую разность $f(x_1) - f(x_2)$. Абсолютная погрешность должна составлять не более 10^{-5}.

Примеры

стандартный ввод	*стандартный вывод*
1 2 3 4	-1.5
2 1 4 3	3

Задача N. Сумма значений функции

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Найдите сумму значений функции

$$f(x) = x + \frac{1}{x}$$

в нескольких целых точках.

Формат входных данных

В первой строке задано целое число n — количество точек ($1 \leqslant n \leqslant 50$). В следующей строке заданы n целых чисел x_1, x_2, …, x_n через пробел — точки, значения функции в которых нужно просуммировать ($0 < |x_i| \leqslant 10^9$).

Формат выходных данных

Выведите одно число — сумму значений функции $f(x)$ в заданных точках. Ответ считается правильным, если абсолютная или относительная погрешность не превышает 10^{-9}.

Примеры

стандартный ввод	*стандартный вывод*
3 1 2 3	7.83333333333333
2 1 -1	0

Задача O. Выражение

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Найдите значение выражения $\sqrt{|a/b + c/d + e/f|}$.

Формат входных данных

В первой строке записаны два целых числа a и b. Во второй строке записаны два целых числа c и d. В третьей строке записаны два целых числа e и f. Здесь $|a|, |c|, |e| \leqslant 10\,000$ и $1 \leqslant b, d, f \leqslant 10\,000$.

Формат выходных данных

Выведите одно вещественное число: значение выражения $\sqrt{|a/b + c/d + e/f|}$. Абсолютная или относительная погрешность должна быть не больше 10^{-9}.

Примеры

стандартный ввод	*стандартный вывод*
1 1 1 1 1 1	1.732050807568877
6 2 -4 3 14 6	2

Задача P. Криптовалютный обменник

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Аля коллекционирует криптовалюты. В её криптовалютном кошельке уже десятки видов валют. Конечно, сейчас каждой валюты у неё немного, но Аля надеется разбогатеть, если какая-то из её криптовалют резко увеличится в цене.

Сегодня Аля нашла в интернете новый криптовалютный обменник — сайт, на котором можно поменять одни криптовалюты на другие. Этот обменник поддерживает n видов криптовалют. Принцип работы обменника прост. Для каждой пары валют указано вещественное число — курс обмена одной валюты на другую. Например, если курс обмена первой валюты на вторую равен 0.585, то за каждую единицу первой валюты можно получить 0.585 единиц второй. В этой задаче мы будем считать, что количество каждой представленной валюты может выражаться любым неотрицательным вещественным числом, и количество исходной валюты при обмене также может быть любым — то есть можно, например, имея 0.1 первой валюты, обменять это количество по тому же курсу на 0.0585 второй.

У Али есть некоторое количество каждой из поддерживаемых валют. Теперь ей интересно, насколько хорошо согласованы курсы обмена: нельзя ли, несколько раз поменяв одну валюту на другую, получить больше исходной криптовалюты, чем было изначально? Помогите ей найти такую цепочку обменов, состоящую из различных валют, или выясните, что её не существует.

Формат входных данных

В первой строке записано целое число n — количество видов криптовалют, которые поддерживает обменник ($2 \leqslant n \leqslant 5$). В каждой из следующих n строк записаны n вещественных чисел: в s-й строке t-е число обозначает курс обмена s-й валюты на t-ю. Все эти числа заданы как десятичные дроби ровно с тремя знаками после десятичной точки, все они строго больше 0.1 и строго меньше 10. Кроме того, гарантируется, что все числа на главной диагонали (курсы обмена криптовалюты на саму себя) равны единице.

Формат выходных данных

Если существует цепочка обменов, после которой можно получить больше исходной криптовалюты, чем её было изначально, выведите в первой строке слово «YES». Во второй строке выведите последовательность чисел c_1, c_2, …, c_k, разделяя соседние числа пробелами. Все эти числа должны быть различны. Количество чисел k может быть любым от 2 до n. Кроме того, при обмене любого количества валюты номер c_1 на валюту номер c_2, затем полученного количества валюты номер c_2 на валюту номер c_3 (если $k \geqslant 3$), … и, наконец, полученного количества валюты номер c_k на валюту номер c_1 итоговое количество валюты номер c_1 должно оказаться строго больше, чем исходное.

Криптовалюты нумеруются с единицы в порядке, в котором они следуют во входных данных. Если существует несколько таких цепочек, можно вывести любую из них.

Если же такой цепочки обменов не существует, выведите в первой строке слово «NO».

Примеры

стандартный ввод	*стандартный вывод*	Пояснение
3 1.000 0.490 0.332 1.990 1.000 0.627 2.985 1.429 1.000	NO	
3 1.000 0.490 0.332 1.990 1.000 0.687 2.985 1.429 1.000	YES 2 3 1	$0.687 \cdot 2.985 \cdot 0.490 =$ $= 1.004\,840\,550 > 1$

Задача Q. Рэп-баттл

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Близится пора школьных выпускных. Два одиннадцатиклассника Петя и Вася хотят пригласить на бал самую красивую девочку параллели Марину. Петя и Вася решили, что Марину на бал должен пригласить самый достойный из них. Выяснить достойнейшего претендента было решено при помощи рэп-баттла по следующим правилам.

К назначенному времени Петя и Вася должны собрать две команды. Команды выстраиваются в две колонны напротив друг друга. Рэп-баттл состоит из раундов. В каждом раунде участники, стоящие во главе колонны, зачитывают свои панчи, после чего судья определяет победителя раунда. Проигравший участник навсегда покидает соревнование, его заменяет следующий участник из колонны, а победитель остаётся на месте. Как только все участники какой-то команды покинули соревнование, эта команда объявляется проигравшей.

Самое главное в рэп-баттле — это красноречие. Если в раунде соревнуются участники со степенью красноречия x и y, то первый выигрывает с вероятностью $\frac{x}{x+y}$.

Вам даны степени красноречия всех членов команд Пети и Васи в порядке их построения. Ваша задача — определить вероятность победы команды Пети.

Формат входных данных

В первой строке через пробел даны два целых числа n и m ($1 \leqslant n, m \leqslant 10\,000$) — число участников в командах Пети и Васи соответственно.

Во второй строке через пробел заданы n целых чисел a_i ($1 \leqslant a_i \leqslant 10\,000$) — степень красноречия i-го участника в команде Пети.

В третьей строке через пробел заданы m целых чисел b_j ($1 \leqslant b_j \leqslant 10\,000$ — степень красноречия j-го участника в команде Васи.

Формат выходных данных

Выведите единственное число: вероятность победы команды Пети с точностью до шести знаков после запятой. Ответ будет считаться верным, если его абсолютная или относительная ошибка не будет превосходить 10^{-6}.

Формально, пусть ваш ответ равен a, а ответ жюри — b. Ваш ответ считается правильным, если $\frac{|a-b|}{\max(1,|b|)} \leqslant 10^{-6}$.

Примеры

стандартный ввод	*стандартный вывод*
1 1 5 3	0.625000
2 2 1 2 1 2	0.500000
5 5 1 2 3 4 5 5 4 3 2 1	0.500000

Пояснения к примерам

В первом примере вероятность того, что игрок из первой команды победит игрока из второй, равна $\frac{5}{5+3} = 0.625$.

Во втором примере победы обеих команд равновероятны, так как команды имеют одинаковые силы и расстановки.

Задача R. Линейные уравнения

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мебибайт

Система линейных уравнений, как всем известно, есть множество уравнений

$$\begin{aligned} a_{11}x_1 + \ldots + a_{1n}x_n &= b_1 \\ &\ldots \\ a_{n1}x_1 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

Ваша задача — решить её.

Формат входных данных

В первой строке записано целое число n ($1 \leqslant n \leqslant 20$). В следующих n строках записано по $n+1$ целых чисел: $a_{i1}, \ldots, a_{in}, b_i$. Все эти числа не превышают 100 по абсолютному значению.

Формат выходных данных

Первая строка должна содержать одно из следующих сообщений:

- `impossible` — решений нет
- `infinity` — бесконечно много решений
- `single` — единственное решение. В этом случае вторая строка должна содержать n чисел $x_1, \ldots, x_n$, разделённых пробелами. Решение должно быть выведено ровно с тремя знаками после десятичной точки.

Примеры

стандартный ввод	*стандартный вывод*
2 1 1 1 2 2 2	infinity
2 1 2 0 1 2 1	impossible
2 1 2 1 2 1 0	single -0.333 0.667

Замечание

Если решение получает неправильный ответ на тесте 34, это проблемы с точностью.