

## Задача А. Линейные уравнения

Имя входного файла: `linear.in`  
Имя выходного файла: `linear.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 64 мегабайта

Система линейных уравнений, как всем известно, есть множество уравнений

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

Ваша задача — решить её.

### Формат входных данных

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

### Формат выходных данных

Первая строка выходного файла 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  $x_1, \dots, x_n$ , разделенных пробелами. Решение должно быть выведено с точностью не менее трех знаков после десятичной точки.

### Примеры

<code>linear.in</code>	<code>linear.out</code>
2 1 1 1 2 2 2	<code>infinity</code>
2 1 2 0 1 2 1	<code>impossible</code>

## Задача В. Обращение матрицы.

Имя входного файла: `inverse.in`  
Имя выходного файла: `inverse.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Дана квадратная матрица. Найдите обратную к ней.

### Формат входных данных

На первой строке входного файла находится одно число  $N$  — размер матрицы ( $1 \leq N \leq 100$ ). Далее следуют  $N$  строк по  $N$  вещественных чисел в каждой — матрица.

### Формат выходных данных

Если обратной матрицы не существует, то выведите в выходной файл одну строку `NO`. Иначе в первой строке выходного файла выведите одно слово `YES`, а далее выведите  $N$  строк по  $N$  вещественных чисел в каждой — обратную матрицу. Ответ будет считаться правильным, если абсолютная или относительная погрешность элементов произведения будет не больше, чем  $10^{-6}$

### Примеры

<code>inverse.in</code>	<code>inverse.out</code>
2 0 1.0 1 0	YES 0.00000000000000000000 1.00000000000000000000 1.00000000000000000000 0.00000000000000000000
2 0 0 1 0	NO

## Задача С. Математизация

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 5 секунд  
Ограничение по памяти: 256 мегабайт

Наступила ночь, но ваш верный слуга не спит и продолжает пытаться выучить теорию следующей лекции. На этот раз лекция будет математической, и Андрею нужно решить очередную задачу. Дано  $n$  натуральных чисел  $a_1, a_2, \dots, a_n$ . Нужно реализовать две операции:

1)  $1\ l\ r\ x$  - прибавить натуральное  $x$  числам на отрезке с  $a_l$  по  $a_r$ .

2)  $2\ l\ r$  - найти сумму  $f(a_l) + f(a_{l+1}) + \dots + f(a_r)$ , где  $f(x)$  - это  $x$ -е число фибоначчи, но сумма может оказаться слишком большой для возможностей мозга Андрея, так что нужно вывести его по модулю  $10^9 + 7$ .

Считаем, что  $f(1) = 1, f(2) = 1, f(x) = f(x-1) + f(x-2)$ , для  $x > 2$ . Помогите Андрею ответить на вопросы задачи.

### Формат входных данных

В первой строке содержатся два числа  $n$  и  $m$  ( $1 \leq n \leq 100000, 1 \leq m \leq 100000$ ) - количество чисел и количество запросов соответственно.

В следующей строке содержатся  $n$  целых чисел  $a_1, a_2, \dots, a_n$ .

Следующие  $m$  строк описывают запросы. В каждой из них находятся числа  $tp_i, l_i, r_i$  и, возможно,  $x_i$  ( $1 \leq tp_i \leq 2, 1 \leq l_i \leq r_i \leq n, 1 \leq x_i \leq 10^9$ ), ( $tp_i = 1$  соответствует запросу первого типа,  $tp_i = 2$  - второго).

Гарантируется, что во входных данных будет присутствовать хотя бы один запрос второго типа.

### Формат выходных данных

Для каждого запроса второго типа в отдельной строке выведите ответ на запрос по модулю  $10^9 + 7$ .

### Пример

стандартный ввод	стандартный вывод
5 4	5
1 1 2 1 1	7
2 1 5	9
1 2 4 2	
2 2 4	
2 1 5	

## Задача D. Число остовных деревьев

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Дан связный неориентированный граф. Найдите число его остовных деревьев по модулю  $10^9 + 7$ .

### Формат входных данных

Первая строка входного файла содержит натуральные числа  $n$  и  $m$  — количество вершин и ребер графа ( $1 \leq n \leq 100$ ,  $1 \leq m \leq 1000$ ). Далее следуют  $m$  строк, задающих ребра. Граф не содержит кратных ребер и петель.

### Формат выходных данных

Выведите число остовных деревьев по модулю  $10^9 + 7$ .

### Примеры

стандартный ввод	стандартный вывод
4 4 2 1 4 2 2 3 4 3	3

## Задача Е. Максимизировать сумму XOR

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 3 секунды  
Ограничение по памяти: 512 мегабайт

Будем обозначать как  $\oplus$  операцию *побитового «исключающего или»* для целых чисел. В языках программирования C++ и Java она обозначается символом «^», в паскале и Python — ключевым словом «xor». Например,  $9 \oplus 3 = 1001_2 \oplus 11_2 = 1010_2 = 10$ .

Даны два массива  $A$  и  $B$  длины  $n$ . Обозначим как  $X(A)$  для массива  $A$  результат вычисления побитового «исключающего или» от всех элементов массива:  $X(A) = A_1 \oplus A_2 \oplus \dots \oplus A_n$ . Аналогично, введем обозначение  $X(B) = B_1 \oplus B_2 \oplus \dots \oplus B_n$ .

Для каждого  $i$  от 1 до  $n$  разрешается поменять местами элементы  $A_i$  и  $B_i$ . Необходимо определить, какие из этих обменов надо сделать, чтобы максимизировать сумму  $X(A) + X(B)$ .

### Формат входных данных

В первой строке входных данных находится число  $n$  — количество элементов ( $1 \leq n \leq 10^5$ ). В следующей строке находится  $n$  элементов массива  $A$  ( $0 \leq A_i \leq 10^{18}$ ). В следующей строке в таком же формате дан массив  $B$ .

### Формат выходных данных

В первой строке выведите максимальную возможную сумму и число  $k$  — количество необходимых обменов. В следующей строке выведите  $k$  различных чисел от 1 до  $n$  — индексы элементов, которые надо поменять.

### Пример

стандартный ввод	стандартный вывод
2	6 1
1 1	1
2 2	

### Замечание

В примере после обмена массивы равны  $A = [2, 1]$  и  $B = [1, 2]$ , соответственно.  
 $X(A) = 2 \oplus 1 = 10_2 \oplus 1_2 = 11_2 = 3$ ,  $X(B) = 3$ ,  $X(A) + X(B) = 6$ .

## Задача F. Квадратное уравнение

Имя входного файла: `quadratic.in`  
Имя выходного файла: `quadratic.out`  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

Детей в школе учат решать квадратные уравнения, т.е. уравнения вида

$$ax^2 + bx + c = 0,$$

где  $a$ ,  $b$  и  $c$  некоторые заданные действительные числа, а  $x$  — действительное число, которое необходимо найти.

В этой задаче вам потребуется решить квадратное уравнение для многочленов с коэффициентами из нулей или единиц, и все операции производятся по модулю 2.

Даны многочлены  $a(t)$ ,  $b(t)$  и  $c(t)$ , найдите такой полином  $x(t)$  что

$$a(t)x^2(t) + b(t)x(t) + c(t) = 0,$$

где равенство понимается как равенство многочленов. Напомним, что многочлены равны тогда и только тогда, когда равны их коэффициенты при соответствующих степенях  $t$ .

### Формат входных данных

Входной файл содержит многочлены  $a(t)$ ,  $b(t)$  и  $c(t)$ , которые задаются их степенями, за которыми следуют коэффициенты, начиная со старшего. Нулевые многочлены в данной задаче имеют степень  $-1$ . Степени всех многочленов не превосходят 127. Между старшим коэффициентом и степенью находится два пробела. После многочлена степени  $-1$  также находится один пробел.

### Формат выходных данных

Если есть хотя бы одно решение уравнения, выведите любое из них в таком же формате. Старший коэффициент найденного многочлена не должен быть нулевым. Степень полинома не должна превышать 512.

В противном случае напечатайте `no solution`.

### Примеры

quadratic.in	quadratic.out
0 1	1 1 0
2 1 1 0	
3 1 0 0 0	

## Задача G. Добавление векторов

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 256 мегабайт

В изначально пустое множество один за другим добавляются  $m$  битовых векторов размера  $n$ . После добавления каждого скажите, можно ли его представить как хог векторов, добавленных до него.

### Формат входных данных

В первой строке записаны числа  $n$  и  $m$  ( $1 \leq n \leq 50$ ,  $1 \leq m \leq 10000$ ). В следующих  $m$  строках записаны вектора.

### Формат выходных данных

Для каждого вектора выведите 1, если его можно представить как хог предыдущих и 0 если нельзя.

### Примеры

стандартный ввод	стандартный вывод
2 2 10 11	0 0
3 4 100 111 011 010	0 0 1 0
3 4 000 111 111 111	1 0 1 1