Задача A. Сортировка небольшой последовательности

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

В этой задаче нужно отсортировать числа, заданные во входных данных.

Формат входных данных

В первой строке входных данных задано целое число n ($1 \leqslant n \leqslant 5000$). Во второй строке заданы через пробел n целых чисел a_1, a_2, …, a_n ($1 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

В первой строке выведите через пробел n чисел — числа a_i в неубывающем порядке.

Примеры

bubble.in	bubble.out
3 1 2 3	1 2 3
4 3 2 2 1	1 2 2 3
5 10 100 10 1000 10	10 10 10 100 1000

Задача B. Белоснежка и n гномов

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

«Ну не гномы, а наказание какое-то!» — подумала Белоснежка, в очередной раз пытаясь уложить гномов спать. Одного уложишь — другой уже проснулся! И так всю ночь.

У Белоснежки n гномов, и все они очень разные. Она знает, что для того, чтобы уложить спать i-го гнома, нужно a_i минут, и после этого он будет спать ровно b_i минут. Помогите Белоснежке узнать, может ли она получить хотя бы минутку отдыха, когда все гномы будут спать, и если да, то в каком порядке для этого нужно укладывать гномов спать.

Например, пусть есть всего два гнома, $a_1 = 1$, $b_1 = 10$, $a_2 = 10$, $b_2 = 20$. Если Белоснежка сначала начнёт укладывать первого гнома, то потом ей потребуется целых 10 минут, чтобы уложить второго, а за это время проснётся первый. Если же она начнёт со второго гнома, то затем она успеет уложить первого и получит целых 9 минут отдыха.

Формат входных данных

Первая строка входных данных содержит число n ($1 \leqslant n \leqslant 10^5$). Вторая строка содержит числа $a_1, a_2, \ldots, a_n$, третья — числа $b_1, b_2, \ldots, b_n$ ($1 \leqslant a_i, b_i \leqslant 10^9$).

Формат выходных данных

Выведите n чисел — порядок, в котором нужно укладывать гномов спать, чтобы получить хотя бы минутку отдыха. Если возможных ответов несколько, выведите любой из них. Если Белоснежке отдохнуть не удастся, выведите число -1.

Примеры

dwarfs.in	dwarfs.out
2 1 10 10 20	2 1
2 10 10 10 10	-1
3 1 4 1 5 3 4	2 1 3

Задача C. Поиск

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

В этой задаче нужно уметь выяснять, содержится ли число в последовательности.

Формат входных данных

В первой строке заданы через пробел два целых числа n и k ($1 \leqslant n \leqslant 5000$, $1 \leqslant k \leqslant 300\,000$). Во второй строке задана последовательность из n целых чисел a_1, a_2, …, a_n, записанных через пробел ($1 \leqslant a_i \leqslant 10^9$). В третьей строке записаны запросы — k целых чисел b_1, b_2, …, b_k через пробел ($1 \leqslant b_j \leqslant 10^9$).

Формат выходных данных

Выведите k строк. В j-й строке выведите «`YES`», если число b_j содержится в последовательности $\{a_i\}$, и «`NO`» в противном случае.

Примеры

find.in	find.out
3 3 2 3 5 1 2 3	NO YES YES
3 4 2 1 2 2 4 1 5	YES NO YES NO
5 1 11111 1111 111 11 1 12345	NO

Задача D. Количество инверсий

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

Напишите программу, которая для заданного массива $A = \langle a_1, a_2, \ldots, a_n \rangle$ находит количество пар (i, j) таких, что $i < j$ и $a_i > a_j$.

Формат входных данных

Первая строка входных данных содержит натуральное число n ($1 \leqslant n \leqslant 50\,000$) — количество элементов массива. Вторая строка содержит n попарно различных элементов массива A.

Формат выходных данных

Выведите одно число — ответ на задачу.

Примеры

inverse.in	inverse.out
4 1 2 4 5	0
4 5 4 2 1	6

Задача E. Медиана последовательности

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

В теории вероятностей и статистике часто рассматриваются конечные последовательности чисел. Иногда для такой последовательности требуется оценить среднее значение её членов. В разных случаях под средним значением понимают разные числа: например, мы можем вычислить среднее арифметическое, а можем посчитать, какое число в отсортированной последовательности будет посередине.

Медианой последовательности называется число, которое может разбить все числа последовательности на два множества: числа в первом множестве не больше медианы, а во втором — не меньше, и при этом мощности множеств равны (то есть в них одинаковое количество элементов). Например, медиана последовательности $\{1, 2, 7, 5, 3\}$ — это 3.

Когда в последовательности нечётное количество членов, медиана определяется однозначно — это тот член последовательности, который находится на равном расстоянии от концов последовательности.

В случае, когда в последовательности чётное количество членов, медианой могло бы служить любое число между двумя средними значениями в последовательности. Например, в последовательности $\{2, 2, 3, 10\,000\}$ медианой могло бы быть любое число из интервала $(2, 3)$. Для определённости в случае чётного количества членов медианой считается среднее арифметическое двух средних значений.

Ваша задача — для данной последовательности целых чисел вывести её медиану.

Формат входных данных

В первой строке входных данных задано целое число n ($1 \leqslant n \leqslant 5000$). Во второй строке заданы через пробел n целых чисел a_1, a_2, …, a_n ($1 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

В первой строке выведите единственное вещественное число — медиану последовательности с точностью до шести знаков после запятой.

Примеры

median.in	median.out
5 1 2 5 7 3	3
4 2 2 3 10000	2.5
5 10 100 10 1000 10	10

Задача F. Минимаксное преобразование

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

Рассмотрим набор чисел a_1, a_2, …, a_n. *Минимаксным преобразованием* этого набора по модулю p назовём следующую операцию: найдём минимум $a_{\min}$ и максимум $a_{\max}$ в наборе, удалим их из него, а вместо них добавим число $(a_{\min} + a_{\max}) \bmod p$.

Дан набор из n чисел. Над ним последовательно $n - 1$ раз производят минимаксное преобразование. Выведите те числа, которые были добавлены в набор, в порядке их добавления.

Формат входных данных

Набор чисел задан генератором. В первой строке входных данных заданы через пробел три целых числа n, b и p ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant b < p \leqslant 10^9$). Числа исходного набора получаются как $a_i = b^i \bmod p$.

Формат выходных данных

В первой строке выведите $n - 1$ число — те числа, которые были добавлены в набор в результате минимаксных преобразований, в порядке их добавления.

Примеры

minimax.in	minimax.out
3 2 5	1 4
4 2 1000	18 22 30

Задача G. Порядковые статистики

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

В этой задаче требуется уметь быстро находить *k-ю порядковую статистику последовательности* — то есть элемент, который после сортировки этой последовательности по неубыванию будет стоять в ней на k-м месте.

Есть две замкнутые в кольцо ленты, на каждой из которых записана последовательность чисел: числа a_1, a_2, …, a_n на первой ленте и числа b_1, b_2, …, b_n на второй. Назовём *p-м циклическим сдвигом* положение, в котором a_1 находится над b_{p+1}, a_2 над b_{p+2}, …, a_{n-p} над b_n, a_{n-p+1} над b_1, …, a_{n-1} над b_{p-1} и a_n над b_p. Рассмотрим последовательность дробей

$$\frac{a_1}{b_{p+1}}, \frac{a_2}{b_{p+2}}, \ldots, \frac{a_{n-p}}{b_n}, \frac{a_{n-p+1}}{b_1}, \ldots, \frac{a_{n-1}}{b_{p-1}}, \frac{a_n}{b_p}$$

и обозначим k-ю порядковую статистику этой последовательности как $s_k^{(p)}$.

По данным числам на лентах, а также числу k, найдите последовательность $s_k^{(1)}$, $s_k^{(2)}$, …, $s_k^{(n)}$.

Формат входных данных

В первой строке входных данных заданы через пробел два целых числа n и k — размер лент и номер порядковой статистики ($1 \leqslant k \leqslant n \leqslant 5000$). Во второй строке записаны n чисел a_1, a_2, …, a_n — содержимое ленты с числителями ($1 \leqslant a_i \leqslant 10^9$). В третьей строке записаны n чисел b_1, b_2, …, b_n — содержимое ленты со знаменателями ($1 \leqslant b_i \leqslant 10^9$).

Формат выходных данных

В первой строке выведите n чисел через пробел — k-е порядковые статистики $s_k^{(1)}$, $s_k^{(2)}$, …, $s_k^{(n)}$. Каждое число должно быть представлено в виде несократимой дроби.

Примеры

orderstatistic.in	orderstatistic.out
3 1 1 2 3 1 2 3	1/2 1/3 1/1
2 2 4 9 4 2	9/4 9/2
3 2 2 2 2 3 4 5	1/2 1/2 1/2

Задача H. Устойчивая сортировка пар

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

В этой задаче нужно отсортировать пары чисел, заданных во входных данных. Зададим *устойчивый порядок* на парах следующим образом. Пусть (a_i, b_i) и (a_j, b_j) — две пары из исходной последовательности, i, j — номера этих пар.

Пара (a_i, b_i) считается меньше пары (a_j, b_j) в двух случаях:

1. если $a_i < a_j$,
2. если $a_i = a_j$ и $i < j$.

Таким образом, требуется отсортировать последовательность пар так, чтобы первые числа пар были расположены в неубывающем порядке, и при этом сортировка была устойчивой.

Формат входных данных

В первой строке входных данных задано целое число n ($1 \leqslant n \leqslant 5000$). В следующих n строках заданы через пробел по два целых числа, a_i и b_i ($1 \leqslant a_i, b_i \leqslant 10^9$).

Формат выходных данных

В первых n строках выведите пары чисел $a_i\, b_i$, разделенных пробелом, в требуемом порядке.

Примеры

pairs.in	pairs.out
3 1 2 3 1 1 1	1 2 1 1 3 1
4 4 1 3 2 2 2 3 3	2 2 3 2 3 3 4 1

Задача I. Сортировка различных чисел

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

В этой задаче нужно отсортировать последовательность натуральных чисел, заданных во входных данных, при условии, что числа в последовательности не повторяются, и максимальное из этих чисел не превосходит длины последовательности.

Формат входных данных

В первой строке входных данных задано натуральное число n ($1 \leqslant n \leqslant 1\,000\,000$). Во второй строке заданы через пробел n натуральных чисел a_1, a_2, …, a_n ($1 \leqslant a_i \leqslant n$).

Формат выходных данных

В первой строке выведите через пробел n чисел — числа a_i в неубывающем порядке.

Пример

permut.in	**permut.out**
2 2 1	1 2

Задача J. Сортировка на плоскости

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

Это интерактивная задача.

Есть n векторов на плоскости, ненулевых и попарно неколлинеарных. Вектор с номером i идёт из начала координат в точку (x_i, y_i). Но эти координаты мы вам не скажем.

Вместо этого вы можете задавать вопросы следующего вида: «Верно ли, что векторы i и j образуют правую пару?» Формально векторы образуют правую пару, если $x_i \cdot y_j > x_j \cdot y_i$. Геометрически правая пара означает, что, если мы стоим в начале координат и смотрим на конец вектора i, то, чтобы как можно скорее повернуться к концу вектора j, следует поворачиваться против часовой стрелки.

Нужно, задавая жюри вопросы, прийти к одному из двух выводов:

1. Все векторы лежат в одной полуплоскости, граница которой проходит через начало координат. Тогда нужно отсортировать их: вывести такой набор различных индексов $i_1, i_2, \ldots, i_n$, что для любых $p < q$ векторы i_p и i_q образуют правую пару.

2. Нет такой полуплоскости, граница которой проходит через начало координат и в которой лежат все векторы. Тогда нужно доказать это: вывести такую последовательность различных индексов $i_1, i_2, \ldots, i_k$, что каждый вектор, кроме последнего, образует правую пару со следующим, а последний (i_k) образует правую пару с первым (i_1).

Протокол взаимодействия

Сначала вашей программе подаётся в отдельной строке число n — количество векторов ($1 \leqslant n \leqslant 500$). Векторы в каждом тесте зафиксированы заранее, но держатся в секрете.

Затем вы можете делать следующее:

1. Спросить у жюри: «верно ли, что векторы i и j образуют правую пару?»

   Для этого ваша программа должна вывести строку следующего вида: «`? i j`». Индексы должны быть корректными: $1 \leqslant i, j \leqslant n$.

   В ответ программа жюри выдаст в отдельной строке число: 1, если ответ «да», и 0, если ответ «нет».

   Чтобы предотвратить буферизацию вывода, после каждого выведенного вопроса следует вставить команду очистки буфера вывода: например, это может быть `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python.

   Вы можете задать не более 20 000 вопросов.

2. Вывести ответ. В этом случае ваша программа должна вывести две строки.

   Если все векторы лежат в одной полуплоскости, первая строка должна иметь вид «`! YES`», а во второй должны быть выведены через пробел n различных чисел от 1 до n: индексы векторов в порядке сортировки.

   Если такой полуплоскости нет, выведите в первой строке «`! NO`». В начале второй строки выведите k — число векторов в доказательстве, а затем выведите k различных чисел от 1 до n — само доказательство. Если возможных доказательств несколько, выведите любое из них.

   После вывода ответа ваша программа должна сразу корректно завершить работу.

Примеры

стандартный ввод	***стандартный вывод***
3 0 1 0	 ? 1 3 ? 3 2 ? 2 1 ! YES 3 1 2
3 1 0 0	 ? 1 2 ? 3 2 ? 1 3 ! NO 3 1 2 3

Задача K. Раствор

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

Перед химиком Колей стоит непростая задача. Для химического эксперимента ему требуется поместить в колбу объёма w как можно большее количество определённого вещества. Это вещество содержится в n пробирках, наполненных растворами; общий объём раствора в i-й пробирке равен q_i, а количество нужного вещества во всём этом растворе — d_i. Считается, что вещество в пробирке равномерно распределено по раствору.

Из каждой пробирки Коля может перелить всё содержимое, или любую его часть, в колбу; различные растворы в колбе смешиваются, а количество вещества и общий объём раствора в колбе складываются. Общий объём раствора, получающегося в колбе, не должен превышать объём колбы w; Коля не может переливать растворы, кроме как из пробирок в колбу. Выясните, какое максимальное количество вещества можно поместить в колбу.

Формат входных данных

В первой строке входных данных заданы через пробел два целых числа n и w — количество пробирок и объём колбы, соответственно ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant w \leqslant 10^9$). Следующие n строк содержат параметры пробирок; i-я из этих строк описывает i-ю пробирку двумя целыми числами d_i и q_i — это количество вещества и общий объём пробирки, соответственно ($1 \leqslant d_i, q_i \leqslant 10^9$). Для удобства записи количество вещества измеряется в одних условных единицах, а объём колбы и общий объём пробирок — в других.

Формат выходных данных

В первой строке выведите максимальное возможное количество вещества в полученном растворе. При выводе представьте это количество в виде смешанной дроби в формате «a+b/c», где a, b и c — целые числа, $a \geqslant 0$, $0 \leqslant b < c$ и $c > 1$. Числа b и c должны быть взаимно просты. Если $a = 0$, то запись должна выглядеть как «b/c». Если $b = 0$, то запись должна состоять только из числа «a».

Примеры

solution.in	solution.out
2 5 3 2 2 5	4+1/5
1 2 2 5	4/5
1 3 3 2	3

Задача L. Сортировка большой последовательности

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

В этой задаче нужно отсортировать числа, заданные во входных данных.

Формат входных данных

В первой строке задано целое число n ($1 \leqslant n \leqslant 300\,000$). Во второй строке заданы через пробел n целых чисел a_1, a_2, …, a_n ($1 \leqslant a_i \leqslant 1000$).

Формат выходных данных

В первой строке выведите через пробел n чисел — числа a_i в неубывающем порядке.

Примеры

sort.in	sort.out
3 1 2 3	1 2 3
4 3 2 2 1	1 2 2 3
5 10 100 10 1000 10	10 10 10 100 1000

Задача M. Сумма двух

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

Дана последовательность чисел, упорядоченная по возрастанию. Выясните, представимо ли число x в виде суммы двух чисел из этой последовательности.

Формат входных данных

В первой строке заданы через пробел два целых числа n и x ($1 \leqslant n \leqslant 100\,000$, $2 \leqslant x \leqslant 2 \cdot 10^9$). Во второй строке задана последовательность из n целых чисел a_1, a_2, …, a_n, записанных через пробел ($1 \leqslant a_i \leqslant 10^9$, все элементы различны, последовательность упорядочена по возрастанию).

Формат выходных данных

Выведите «`YES`», если число x представимо в виде $a_i + a_j$ для некоторых (не обязательно различных) индексов i и j, и «`NO`» в противном случае.

Примеры

sumtwo.in	sumtwo.out
3 4 1 2 3	YES
3 5 1 2 5	NO
4 6 1 2 4 8	YES

Задача N. Ленивый работник

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

Трактир «Не дай себе просохнуть!» взял на работу нового работника — кладовщика. У кладовщика есть две обязанности. Первая — принимать от грузчиков бочки с пивом и закатывать их в подвал. Вторая — когда в бочке у трактирщика не остаётся пива, выкатывать ему из подвала новую бочку, в которой пиво ещё имеется.

Хитрый работник скоро понял, что в разных бочках, попадающих к нему в подвал, содержится различное количество пива, и решил впредь всегда выкатывать трактирщику ту бочку, в которой пива менее всего, чтобы особо не напрягаться. Дальновидность не в характере лентяя-кладовщика, и то обстоятельство, что позднее ему придётся катать исключительно тяжёлые бочки, равно как и то, что лёгкие бочки нужно катать чаще, он проигнорировал.

При получении бочки от грузчиков кладовщик измеряет количество пива в ней. Количество пива обозначается целым числом от 0, когда бочка пуста, до 1000, когда вся бочка заполнена пивом. Известно, что грузчики не передают кладовщику пустых бочек.

Трактирщик узнал о том, какую хитрость придумал его новый ленивый работник, и теперь, зная количество пива во всех привозимых грузчиками бочках, хочет выяснить, сколько пива будет в каждой следующей бочке, которую кладовщик выкатит из подвала.

По последовательности поступлений бочек в подвал и выкатываний их трактирщику установите, сколько пива было в каждой выкаченной бочке.

Формат входных данных

В первой строке входных данных задано число N ($1 \leqslant N \leqslant 100\,000$). Следующие N строк содержат по одному числу A_i в каждой и описывают события в порядке, в котором они происходят. Если A_i больше нуля, то это означает поступление в подвал новой бочки, количество пива в которой равно A_i. Если же A_i равно нулю, то описываемое данной строкой событие — это выкатывание из подвала бочки, пива в которой осталось менее всего.

Изначально можно считать, что в подвале бочек с пивом нет. Гарантируется, что трактирщик не попадёт в ситуацию, когда на его запрос кладовщик не может выкатить новой бочки.

Формат выходных данных

На каждое выкатывание бочки из подвала выведите строку, состоящую из единственного числа B_j — сколько пива содержалось в выкаченной бочке.

Примеры

worker.in	worker.out
3 1 2 0	1
5 3 0 2 0 1	3 2
6 8 4 2 1 0 0	1 2
4 1 1 1 0	1
10 4 3 7 4 9 5 6 0 0 0	3 4 4