Задача A. Количество инверсий

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

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 входного файла

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 50\,000$) — количество элементов массива. Вторая строка содержит n попарно различных элементов массива A — целых неотрицательных чисел, не превосходящих 10^6.

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Примеры

inverse.in	inverse.out
5 6 11 18 28 31	0
8 999994 999989 999982 999972 999969 999961 999954 999950	28

Задача B. Мутанты

Имя входного файла:	`mutants.in`
Имя выходного файла:	`mutants.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Уже долгое время в Институте Искусств, Мутантов и Информационных Технологий разводят милых разноцветных зверюшек. Для удобства каждый цвет обозначен своим номером, всего цветов не более 10^9. В один из прекрасных дней в питомнике случилось чудо: все зверюшки выстроились в ряд в порядке возрастания цветов. Пользуясь случаем, лаборанты решили посчитать, сколько зверюшек разных цветов живет в питомнике, и, по закону жанра, попросили вас написать программу, которая поможет им в решении этой нелегкой задачи.

Формат входного файла

В первой строке входного файла содержится единственное число N ($0 \leqslant N \leqslant 10^6$) — количество зверюшек в Институте. В следующей строке находятся N упорядоченных по неубыванию неотрицательных целых чисел, не превосходящих 10^9 и разделенных пробелами — их цвета. В третьей строке файла записано число M ($0 \leqslant M \leqslant 200\,000$) — количество запросов вашей программе, в следующей строке через пробел записаны M целых неотрицательных чисел (не превышающих $10^9 + 1$).

Формат выходного файла

Выходной файл должен содержать M строчек. Для каждого запроса выведите число зверюшек заданного цвета в питомнике.

Примеры

mutants.in	mutants.out
10 1 1 3 3 5 7 9 18 18 57 5 57 3 9 1 179	1 2 1 2 0

Задача C. Преферанс

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

В новой колоде 32 карты для преферанса расположены в следующем порядке (сверху вниз): червы, бубны, трефы, пики.

В каждой масти сначала лежит семерка, под ней — восьмерка, затем девятка, десятка, валет, дама, король, туз. Тасовка карт осуществляется так: 16 карт, составляющих верхнюю половину колоды, распределяются между картами нижней половины колоды.

Каждая карта верхней половины вставляется в нижнюю колоду так, что в получившейся колоде карты верхней половины идут в том же порядке, в котором они были изначально. Любое число карт верхней половины можно располагать как над верхней, так и под нижней картой второй половины колоды, а также между любыми двумя соседними картами нижней половины колоды. Такие действия повторяются не более пяти раз.

Требуется написать программу, которая указывает, как надо осуществить тасовку, чтобы в итоге получить заранее заданное расположение карт.

Формат входного файла

Единственная строка входного файла содержит информацию о порядке карт, в котором они должны оказаться после тасовки. Карты перечислены сверху вниз.

Каждая карта обозначается латинской буквой, указывающей масть (пики — `S`, трефы — `C`, бубны — `D`, червы — `H`), и номиналом (туз — `A`, король — `K`, дама — `Q`, валет — `J`, десятка — `0`, остальные — в соответствии со своим значением: `9`, `8`, `7`).

Формат выходного файла

В первую строку выходного файла необходимо вывести целое число N ($0 \leqslant N \leqslant 5$) — количество шагов тасовки. Следующие N строк должны содержать информацию о каждом шаге тасовки. Каждая строка при этом должна содержать 16 чисел, указывающих номера позиций, на которых оказываются снятые карты. Номера позиций выводятся в порядке возрастания и разделены пробелами. Нумерация позиций производится сверху вниз от 1 до 32.

Примеры

pref.in	pref.out
C7 H7 C8 H8 C9 H9 C0 H0 S7 D7 S8 D8 S9 D9 S0 D0 SJ DJ SQ DQ SK DK SA DA CJ HJ CQ HQ CK HK CA HA	5 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 1 2 5 6 9 10 13 14 19 20 23 24 27 28 31 32 1 2 3 4 9 10 11 12 17 18 19 20 25 26 27 28 1 2 5 6 9 10 13 14 17 18 21 22 25 26 29 30 1 2 5 6 9 10 13 14 17 18 21 22 25 26 29 30

Задача D. Автомобиль Джона

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

Джон недавно купил новый автомобиль. Он решил показать его всем своим друзьям, побывав на каждой улице города, в котором он живет. А для того, чтобы потратить как можно меньше бензина, он хочет проехать по каждой улице ровно один раз.

Город Джона состоит из улиц и перекрестков. Каждая улица имеет уникальный номер i ($1 \leqslant i \leqslant 1994$). Каждый перекресток также имеет уникальный номер j ($1 \leqslant j \leqslant 44$). Нумерация перекрестков никак не связана с нумерацией улиц. Каждая улица соединяет ровно два перекрестка (могут существовать улицы, соединяющие перекресток с самим собой). Могут быть пары перекрестков, между которыми есть несколько улиц.

Перекресток, на котором живет Джон, находится на конце первой улицы и имеет минимальный номер. Составляя план объезда всех улиц в виде последовательности номеров улиц, Джон выбрал план, соответствующий лексикографически наименьшей последовательности.

Формат входного файла

Входной файл содержит несколько тестовых случаев. Каждый тестовый случай представляет собой описание города. Город описан несколькими строчками, каждая из которых описывает одну из улиц и содержит три числа: x, y и z ($1 \leqslant x, y \leqslant 44$, $1 \leqslant z \leqslant 1994$), где x и y — номера перекрестков, соединенных улицей, а z — номер самой улицы. Каждый тестовый случай заканчивается строкой, содержащей два нуля. Еще одна строка с двумя нулями означает конец входного файла.

Формат выходного файла

Для каждого тестового случая выведите блок из двух строк, содержащий ответ. В первой строке должна быть последовательность номеров улиц в порядке посещения улиц Джоном. Вторая строка должна быть пустой. Если у описанного города не существует ни одного требуемого объезда, то выведите без кавычек фразу «Round trip does not exist.»

Примеры

john.in	john.out
1 2 1 2 3 2 3 1 6 1 2 5 2 3 3 3 1 4 0 0 1 2 1 2 3 2 1 3 3 2 4 4 0 0 0 0	1 2 3 5 4 6 Round trip does not exist.

Задача E. K-й минимум

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

Напишите программу, которая находит k-е в возрастающем порядке число в массиве $A = \langle a_1, a_2, \ldots, a_n \rangle$.

Массив A задается с помощью полинома $P(x) = 132x^3 + 77x^2 + 1345x + 1577$: $a_i = P(i) \bmod 1743$.

Формат входного файла

Входной файл содержит два натуральных числа n и k ($1 \leqslant k \leqslant n \leqslant 50\,000$).

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Примеры

kth.in	kth.out
1 1	1388
10 1	402

Note

Предполагается решение за $O(n)$.

Задача F. K-й минимум — 2

Имя входного файла:	`kth2.in`
Имя выходного файла:	`kth2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Напишите программу, которая находит k-е в возрастающем порядке число в массиве $A = \langle a_1, a_2, \ldots, a_n \rangle$.

Массив A задается с помощью полинома $P(x) = 132x^3 + 77x^2 + 1345x + 1577$: $a_i = P(i) \bmod 1\,000\,000\,003$.

Формат входного файла

Входной файл содержит два натуральных числа n и k ($1 \leqslant k \leqslant n \leqslant 50\,000$).

Формат выходного файла

В выходной файл выведите одно число — ответ на задачу.

Примеры

kth2.in	kth2.out
1 1	3131
10 1	3131

Note

Предполагается решение за $O(n)$.