A. Поиск красивых чисел

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Дано два списка различных ненулевых цифр.

Назовем число красивым, если в его записи (в системе счисления по основанию 10) присутствует хотя бы одна цифра из первого списка и хотя бы одна цифра из второго списка. Чему равно минимальное натуральное (положительное целое) красивое число?

Входные данные

В первой строке даны числа n, m ($1 \le n, m \le 9$) – длины первого и второго списка соответственно.

Во второй строке через пробел даны n различных целых цифр $a_1, a_2, ..., a_n$ ($1 \le a_i \le 9$) – элементы первого списка.

В третьей строке через пробел даны m различных целых цифр $b_1, b_2, ..., b_m$ ($1 \le b_i \le 9$) – элементы второго списка.

Выходные данные

Выведите минимальное натуральное красивое число.

Примеры

входные данные	Скопировать
2 3 4 2 5 7 6	

выходные данные
25

входные данные	Скопировать
8 8 1 2 3 4 5 6 7 8 8 7 6 5 4 3 2 1	

выходные данные
1

Примечание

В первом примере красивыми являются числа $25, 46, 24567$ и многие другие. Из них минимальным является 25. 42 и 24 не являются красивыми, так как в них отсутствуют цифры из второго списка.

Во втором примере красивыми являются все числа, в чьей записи встречаются не только цифры 9. Очевидно, минимальным из таких чисел является 1, так как это минимальное натуральное число.

B. Максимум максимума из минимумов

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Дан массив $a_1, a_2, ..., a_n$ из n чисел, а также число k. Массив нужно разбить на ровно k непустых подотрезков. На каждом подотрезке вычисляется минимум, а из полученных k минимумов берется максимум. А какое максимальное число можно таким образом получить?

Определения подотрезка и разбиения массива на подотрезки находятся в примечаниях.

Входные данные

В первой строке записаны через пробел два целых числа n и k ($1 \le k \le n \le 10^5$) — размер массива a и количество подотрезков, на которые нужно разбить массив.

Во второй строке записано n целых чисел через пробел: $a_1, a_2, ..., a_n$ ($-10^9 \le a_i \le 10^9$).

Выходные данные

Выведите единственное число — максимальный максимум, который можно получить, разбив данный массив на k подотрезков и взяв максимум из минимумов на этих подотрезках.

Примеры

входные данные	Скопировать
5 2 1 2 3 4 5	
выходные данные	
5	

входные данные	Скопировать
5 1 -4 -5 -3 -2 -1	
выходные данные	
-5	

Примечание

Подотрезок $[l, r]$ ($l \le r$) массива a — это последовательность $a_l, a_{l+1}, ..., a_r$.

Разбиение массива a из n элементов на k подотрезков, $[l_1, r_1], [l_2, r_2], ..., [l_k, r_k]$ ($l_1 = 1, r_k = n, l_i = r_{i-1} + 1$ при $i > 1$), — k последовательностей $(a_{l_1}, ..., a_{r_1}), ..., (a_{l_k}, ..., a_{r_k})$.

В первом примере вы должны разбить массив на подотрезки $[1, 4]$ и $[5, 5]$, тогда вы получите последовательности $(1, 2, 3, 4)$ и (5). Минимумы равны $min(1, 2, 3, 4) = 1$ и $min(5) = 5$. Итоговый максимум равен $max(1, 5) = 5$. Очевидно, получить больший результат нельзя.

Во втором примере единственная ваша возможность — разбить массив на единственный подотрезок $[1, 5]$, тогда вы получите последовательность $(-4, -5, -3, -2, -1)$. Единственный минимум равен $min(-4, -5, -3, -2, -1) = -5$. Итоговый максимум равен -5.

C. Максимальное разбиение

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Вам задано некоторое количество запросов. В i-м запросе дано положительное целое число n_i. Надо представить n_i в виде суммы максимального количества составных слагаемых и вывести это количество, либо вывести `-1`, если такое представление невозможно.

Составными называются натуральные числа большие 1, не являющиеся простыми, то есть, имеющие натуральные делители, отличные от 1 и самого числа.

Входные данные

В первой строке дано число q ($1 \le q \le 10^5$) — количество запросов

Далее идет q строк. В $(i+1)$-й строке дано число n_i ($1 \le n_i \le 10^9$) — i-й запрос.

Выходные данные

Для каждого запроса выведите максимальное количество слагаемых в корректном разбиении на составные слагаемые или `-1`, если такого разбиения не существует.

Примеры

входные данные	Скопировать
1 12	

выходные данные
3

входные данные	Скопировать
2 6 8	

выходные данные
1 2

входные данные	Скопировать
3 1 2 3	

выходные данные
-1 -1 -1

Примечание

$12 = 4 + 4 + 4 = 4 + 8 = 6 + 6 = 12$, но в первом разбиении больше количество слагаемых

$8 = 4 + 4$, 6 нельзя разбить на меньшие составные слагаемые.

$1, 2, 3$ меньше любого составного числа, поэтому для них не существует корректных разбиений.

D. Что-то там с xor запросами

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Это интерактивная задача.

Жюри загадало некоторую перестановку p из чисел от 0 до n - 1, про которую вам известна только ее длина n. Напомним, что в перестановке все числа различны.

Пусть b — обратная к p перестановка, то есть $p_{b_i} = i$ для всех i. Тогда единственное действие, которое вы можете выполнить — узнать `xor` элементов p_i и b_j, указав их индексы i и j (не обязательно различные). В результате запроса для индексов i и j вы получите значение $p_i \oplus b_j$, где символом $\oplus$ обозначена операция `xor`. Описание операции `xor` вы можете найти в примечаниях.

Заметим, что некоторые перестановки могут быть неотличимы от заданной, даже если сделать все различные n^2 запросов. Вам необходимо вычислить, сколько перестановок неотличимы от загаданной, а также выдать одну из таких перестановок, сделав не более $2n$ запросов.

Загаданная перестановка не зависит от ваших запросов.

Входные данные
В первой строке входных данных следует целое положительное число n ($1 \le n \le 5000$) — длина загаданной перестановки. В первую очередь ваша программа должна прочитать это число.

Выходные данные
Когда ваша программа будет готова вывести ответ, выведите три строки.

В первой строке выведите один символ «`!`».

Во второй строке выведите одно целое число $answers\_cnt$ — количество перестановок, неотличимых от загаданной жюри, считая загаданную.

В третьей строке выведите n целых чисел $p_0, p_1, ..., p_{n-1}$ ($0 \le p_i < n$, все p_i должны быть различны) — одну из перестановок, неотличимых от загаданной жюри.

После вывода ответа ваша программа должна завершиться.

Протокол взаимодействия
Чтобы узнать `xor` двух элементов, выведите строку вида «`? i j`», где i и j — целые числа от 0 до n - 1 — индекс элемента перестановки и индекс элемента обратной перестановки, `xor`-сумму которых ваша программа запрашивает. После этого выведите перевод строки и сделайте операцию `flush`.

После выполнения запроса ваша программа может считать целое число, равное $p_i \oplus b_j$.

Для перестановки длины n ваша программа должна сделать не более $2n$ запросов на `xor`-сумму. Обратите внимание, что вывод ответа не учитывается при подсчете количества запросов. Обратите внимание, что вы можете задать не более $2n$ запросов. В случае, если ваша программа сделает больше $2n$ запросов или сделает хотя бы один некорректный запрос, ваше решение получит вердикт «`Неправильный ответ`».

Если в какой-то момент ваша программа считывает `-1` как ответ, она должна немедленно завершиться (например, вызовом `exit(0)`). Вы получите вердикт «`Неправильный ответ`», и это будет означать, что вы задали больше $2n$ запросов или задали некорректный запрос. Если вы проигнорируете это, то можете получить любой вердикт, так как ваша программа продолжит читать из закрытого потока ввода.

Выше решение получит вердикт «`Решение зависло`», если вы не будете ничего выводить или забудете сделать операцию `flush` после вывода вопроса или ответа.

Чтобы выполнить операцию `flush`, можете использовать (сразу после вывода запроса и перевода строки):

- `fflush(stdout)` в C++;
- `System.out.flush()` в Java;
- `stdout.flush()` в Python;
- `flush(output)` в Pascal;
- Для других языков смотрите документацию.

Взломы

Используйте следующий формат для взломов:

n

$p_0 p_1 \cdots p_{n-1}$

Взламываемая программа не будет иметь доступа к этим данным.

Примеры

входные данные	Скопировать
3 0 0 3 2 3 2	

выходные данные
? 0 0 ? 1 1 ? 1 2 ? 0 2 ? 2 1 ? 2 0 ! 1 0 1 2

входные данные	Скопировать
4 2 3 2 0 2 3 2 0	

выходные данные
? 0 1 ? 1 2 ? 2 3 ? 3 3 ? 3 2 ? 2 1 ? 1 0 ? 0 0 ! 2 3 1 2 0

Примечание

Операцией `xor`, или побитовое исключающее ИЛИ, называется операция над двумя целыми числами, при которой i-й разряд результата в двоичной системе счисления будет равен 1 тогда и только тогда, когда ровно у одного из двух целых чисел в i-м разряде в двоичной системе счисления стоит 1. Больше информации смотрите по ссылке.

В первом примере при $p = [0, 1, 2]$, а значит $b = [0, 1, 2]$, на заданные запросы у этой перестановки совпадают все значения $p_i \oplus b_j$ для всех выведенных парах i, j. Кроме этой перестановки не существует других перестановок, подходящих под выданные ответы, поэтому это — ответ.

Ответы на запросы:

- $p_0 \oplus b_0 = 0 \oplus 0 = 0$,
- $p_1 \oplus b_1 = 1 \oplus 1 = 0$,
- $p_1 \oplus b_2 = 1 \oplus 2 = 3$,
- $p_0 \oplus b_2 = 0 \oplus 2 = 2$,
- $p_2 \oplus b_1 = 2 \oplus 1 = 3$,
- $p_2 \oplus b_0 = 2 \oplus 0 = 2$.

В втором примере при $p = [3, 1, 2, 0]$, а значит $b = [3, 1, 2, 0]$, на заданные запросы у этой перестановки совпадают все значения $p_i \oplus b_j$ для всех выведенных пар i, j. Однако кроме нее подходит также перестановка $p = [0, 2, 1, 3]$, $b = [0, 2, 1, 3]$, причем на всех n^2 возможных запросов у этих двух перестановок ответы будут совпадать, а у всех остальных перестановок ответы будут другие уже на заданных запросах.

E. Точки, прямые и неоригинальные названия

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

На плоскости дано n различных точек с целыми координатами. Для каждой точки можно выполнить одно из трех действий: провести через нее вертикальную прямую, провести через нее горизонтальную прямую или ничего не делать.

Если рассматривать несколько совпадающих прямых как одну, то сколько различных картинок может получиться? Ответ вывести по модулю $10^9 + 7$.

Входные данные

В первой строке дано целое число n ($1 \le n \le 10^5$) — количество точек.

Далее идет n строк. В $(i + 1)$-й строке даны два целых числа x_i, y_i ($-10^9 \le x_i, y_i \le 10^9$) — координаты i-й точки.

Гарантируется, что все точки различны.

Выходные данные

Вывести количество различных картинок по модулю $10^9 + 7$.

Примеры

входные данные	Скопировать
4 1 1 1 2 2 1 2 2	

выходные данные
16

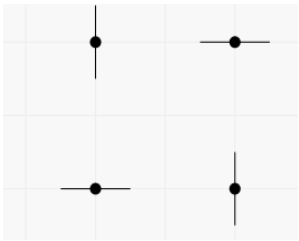
входные данные	Скопировать
2 -1 -1 0 1	

выходные данные
9

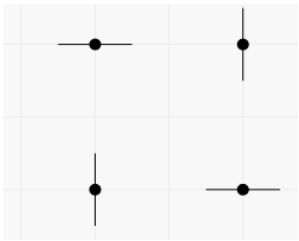
Примечание

В первом примере через точки проходят две вертикальные и две горизонтальные прямые. Существуют способы получить картинку из любого набора этих прямых. В частности, получить картинку на которой нарисованы все четыре прямые можно двумя способами (каждый отрезок обозначает прямую, которой отрезок принадлежит).

Первый способ:



Второй способ:



Во втором примере для каждой точки можно независимо выполнить одно из трех действий. Количество картинок равно $3^2 = 9$.

F. Пути

ограничение по времени на тест: 4 секунды
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Дано положительное целое число n. Построим граф на вершинах $1, 2, ..., n$ так, чтобы ребро между вершинами u и v существовало тогда и только тогда, когда $\gcd(u, v) \neq 1$. Пусть $d(u, v)$ — кратчайшее расстояние между u и v или 0, если между ними нет пути. Посчитайте сумму $d(u, v)$ для всех $1 \leq u < v \leq n$.

gcd или НОД (наибольший общий делитель) двух натуральных чисел — такое наибольшее натуральное число, которое делит оба этих числа нацело.

Входные данные
Целое число n ($1 \leq n \leq 10^7$).

Выходные данные
Выведите сумму $d(u, v)$ для всех $1 \leq u < v \leq n$.

Примеры

входные данные	Скопировать
6	

выходные данные
8

входные данные	Скопировать
10	

выходные данные
44

Примечание
Все кратчайшие пути в первом примере:

- $2 \to 6 \to 3$
- $2 \to 4$
- $2 \to 6$
- $3 \to 6 \to 4$
- $3 \to 6$
- $4 \to 6$

Между остальными парами вершин путь не существует.

Суммарное расстояние $2 + 1 + 1 + 2 + 1 + 1 = 8$.