Задача A. Красивые числа

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

Назовем число *красивым*, если сумма квадратов его цифр в десятичной системе счисления является простым числом. Найдите n в порядке возрастания красивое число.

Напомним, что положительное целое число x называется *простым*, если оно больше единицы и не делится ни на одно из чисел от 2 до $x - 1$.

Формат входного файла

Во входном файле находится положительное целое число n ($1 \le n \le 10000$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

beauty.in	beauty.out
1	11
6	23

Задача B. Циклический класс

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

Перестановкой порядка n называется последовательность из попарно различных целых положительных чисел $p_1, p_2, \ldots, p_n$, где каждое $1 \le p_i \le n$. *Циклом перестановки* называется такая последовательность $i_1, i_2, \ldots, i_k$ различных положительных целых чисел от 1 до n, что $p_{i_1} = i_2$, $p_{i_2} = i_3, \ldots, p_{i_k} = i_1$.

Любую перестановку можно единственным образом разбить на несколько циклов так, чтобы каждый ее элемент принадлежал ровно одному циклу. *Циклическим классом* перестановки называется последовательность, состоящая из длин ее циклов, упорядоченная по неубыванию.

Задана перестановка. Найдите ее циклический класс.

Формат входного файла

Первая строка входного файла содержит порядок n ($1 \le n \le 1000$) заданной перестановки. Вторая строка содержит числа $p_1, p_2, \ldots, p_n$, отделенные друг от друга пробелами.

Формат выходного файла

В первой строке выходного файла выведите количество циклов c в заданной перестановке. Во второй строке выведите длины циклов в порядке неубывания.

Примеры

cyclic.in	cyclic.out
5 3 4 1 2 5	3 1 2 2
5 3 4 2 1 5	2 1 4

Задача C. Перестановки со знаком без неподвижных точек

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

Перестановкой со знаком размера n называется упорядоченный набор из n чисел, каждое из которых находится в диапазоне от $-n$ до n и не равно 0. При этом абсолютные значения всех чисел должны быть различны. Например, $\langle 4, -2, 3, -5, -1 \rangle$ — перестановка со знаком размера 5. Всего существует $2^n n!$ перестановок со знаком размера n.

Говорят, что перестановка со знаком $\langle a_1, a_2, \ldots, a_n \rangle$ не содержит неподвижных точек, если $a_i \neq i$ для всех i (по-английски такие перестановки называются *derangements*). Так, перестановка $\langle 4, -2, 3, -5, -1 \rangle$ содержит неподвижную точку (число 3), а перестановка $\langle 4, -2, -3, -5, -1 \rangle$ — нет.

По заданному n, найдите количество перестановок со знаком размера n, не имеющих неподвижных точек.

Формат входного файла

Входной файл содержит одно число n ($1 \leq n \leq 200$).

Формат выходного файла

Выведите одно число — количество перестановок со знаком размера n, не имеющих неподвижных точек.

Пример

derangements.in	derangements.out
2	5

Следующие перестановки со знаком размера 2 не имеют неподвижных точек: $\langle 2, 1 \rangle$, $\langle 2, -1 \rangle$, $\langle -2, 1 \rangle$, $\langle -1, -2 \rangle$, и $\langle -2, -1 \rangle$.

Задача D. Игра на бирже

Имя входного файла:	`financial.in`
Имя выходного файла:	`financial.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Петя пишет программы для бирж. Он занимается разработкой различных финансовых инструментов, его последний проект связан с анализом возможности арбитража с использованием исторических данных о котировках существенно волатильной акции.

Идея анализа состоит в следующем: по заданным котировкам акции в течение n последовательных дней, требуется выбрать подпоследовательность котировок, которая удовлетворяет следующему свойству: любые две соседние котировки в выбранной подпоследовательности должны отличаться не менее чем на k. Например, если котировки равны, соответственно, $1014, 1024, 1034, 1045, 1030, 998$ и $k = 15$, то подпоследовательность котировок $1014, 1034, 998$ является допустимой.

Выбранная подпоследовательность должны быть как можно длиннее. Так, в приведенном примере выбранная последовательность не является оптимальной, подпоследовательность $1014, 1045, 1030, 998$ лучше.

По заданной последовательности котировок, найдите ее длиннейшую допустимую подпоследовательность.

Формат входного файла

Первая строка входного файла содержит число n ($1 \le n \le 100\,000$) — количество котировок и k ($1 \le k \le 10^9$). Вторая строка содержит n целых чисел — последовательность котировок (все котировки находятся в интервале от 1 до 10^9, включительно).

Формат выходного файла

Первая строка выходного файла должна содержать l — длину оптимальной подпоследовательности. Вторая строка должна содержать l целых чисел — элементы любой такой подпоследовательности.

Пример

financial.in	financial.out
6 15 1014 1024 1034 1045 1030 998	4 1014 1045 1030 998

Задача E. Сумма k подмножеств

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

Одной из классических NP-полных задач является так называемая *задача о сумме подмножества* (SUBSET-SUM problem). Она формулируется следующим образом: «Дано n чисел a_1, a_2, …, a_n. Возможно ли выбрать несколько из них так, чтобы их сумма была равна S?».

Эта задача допускает несколько обобщений. Одно из них мы назовем *задача о сумме k подмножеств* и будем рассматривать далее. Пусть задано n чисел a_1, a_2, …, a_n и k чисел s_1, s_2, …, s_k. Необходимо разбить числа a_1, …, a_n на k непустых наборов, чтобы сумма чисел в первом из них была равна s_1, во втором — s_2, …, в k-ом — s_k. Каждое число может включено не более чем в один набор.

Напишите программу, решающую задачу о сумме k подмножеств.

Формат входного файла

Первая строка входного файла содержит два целых числа n и k ($1 \le n, k \le 10$, $n^k \le 2^{24}$). Вторая строка входного файла содержит n целых чисел a_1, a_2, …, a_n (для всех a_i верно неравенство $|a_i| \le 10^9$). Третья строка входного файла содержит k целых чисел s_1, s_2, …, s_k (для всех s_j верно неравенство $|s_j| \le 10^9$).

Формат выходного файла

Если искомое разбиение на наборы существует, то выведите в первой строке выходного файла слово `YES`, иначе — слово `NO`.

Примеры

ksubset.in	ksubset.out
4 2 1 2 3 4 5 5	YES
4 2 1 2 3 4 1 8	NO

Задача F. Move to Front

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

«Move-to-Front» — название метода преобразования последовательностей целых чисел, которой используется в некоторых алгоритмах сжатия, таких как преобразование Барроуза-Уилера.

Исходно все натуральные числа считаются упорядоченными в возрастающем порядке в список L. Рассмотрим последовательность $a_1, a_2, \ldots, a_n$ натуральных чисел. Она будет закодирована в последовательность $b_1, b_2, \ldots, b_n$ следующим образом. Пусть часть от a_1 до a_{i-1} уже закодирована. Рассмотрим позицию числа a_i в текущем списке L. Эта позиция присваивается числу b_i, а число a_i в списке L перемещается в начало.

Например, последовательность $3, 3, 3, 2, 2, 2, 2, 2, 3, 1, 3, 3, 2$ будет закодирована как $3, 1, 1, 3, 1, 1, 1, 1, 2, 3, 2, 1, 3$.

Вам задана последовательность $a_1, a_2, \ldots, a_n$, требуется закодировать ее с использованием метода «Move-to-Front» и вывести получившуюся последовательность $b_1, b_2, \ldots, b_n$.

Формат входного файла

Первая строка входного файла содержит число n ($1 \le n \le 100\,000$). Вторая строка содержит n целых чисел a_i, каждое из которых от 1 до 10^9.

Формат выходного файла

Выведите n целых чисел — последовательность $b_1, b_2, \ldots, b_n$.

Пример

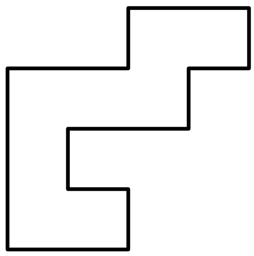
mtf.in	mtf.out
13 3 3 3 2 2 2 2 2 3 1 3 3 2	3 1 1 3 1 1 1 1 2 3 2 1 3

Задача G. Ночной сторож

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

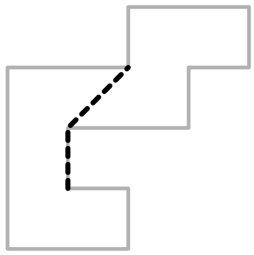
Владислав Владиславович Тупиков работает ночным сторожем в музее. Каждый час он делает обход музея, чтобы проверить, что все в порядке. Поскольку Владиславу Владиславовичу уже 70 лет, ему довольно тяжело ходить, поэтому он хочет, чтобы эти прогулки были как можно короче.

Рассмотрим план музея, введем систему координат так, чтобы стены музея были параллельны осям координат. Музей имеет форму простого монотонного прямоугольного многоугольника: все его углы прямые, а каждая прямая $y = c$ либо не пересекает внутренность музея, либо пересекает ее по отрезку. Пример плана музея приведен на рисунке.



Владислав Владиславович может начать свой путь в любой точке музея, пройти любым способом и вернуться в исходную точку. Помогите ему найти путь минимальной длины, такой, чтобы каждая точка внутри музея была видна хотя бы из одной точки на пути. Точка A считается видимой из точки B, если соединяющий их отрезок лежит внутри музея (он может иметь общие точки с границей музея).

Так, оптимальный путь в приведенном на рисунке сверху музее показан на следующем рисунке (Владислав Владиславович может начать из любой точки и пройти по этому пути туда и обратно).



По заданному плану музея помогите Владиславу Владиславовичу найти оптимальный путь.

Формат входного файла

Первая строка входного файла содержит число n — количество вершин в ломаной, которая представляет собой границу музея на плане ($4 \le n \le 100$). Следующие n строк содержат координаты вершин в порядке их обхода против часовой стрелки. Координаты не превышают 10^4 по модулю.

Формат выходного файла

Выведите одно число — длину оптимального пути. Выведенное число должно отличаться от правильного ответа не более чем на 10^{-5}.

Пример

nightwatch.in	nightwatch.out
12 1 0 3 0 3 1 2 1 2 2 4 2 4 3 5 3 5 4 3 4 3 3 1 3	4.8284271247461901
4 0 0 1 0 1 1 0 1	0.0

Во втором примере сторож может просто сидеть на одном месте и при этом ему видно весь музей.

Задача H. Путь из S в T

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

Вы являетесь одним из работников отдела логистики одной крупной транснациональной корпорации. Недавно ваша корпорация начала осваивать рынок страны N-мерики. Первый филиал открылся в городе S. Ближайшая цель — открыть торговое представительство в городе T. Для этого туда необходимо перевести оборудование суммарной массой M тонн. Оборудование упаковано в контейнер, который запрещается вскрывать до прибытия в город T, поэтому его придется перевозить как единое целое.

Известна карта дорог N-мерики и максимальный тоннаж груза, который разрешено перевозить по каждой из дорог. Необходимо выяснить, можно ли перевезти груз массой M тонн из города S в город T.

Формат входного файла

В первой строке входного файла находятся два целых числа n и m — количество городов и дорог в N-мерике ($2 \le n \le 1000$, $0 \le m \le 100000$). Города пронумерованы натуральными числами от 1 до n, город S имеет номер 1, город T — номер n. Вторая строка входного файла содержит целое число M ($1 \le M \le 10^9$).

Последующие m строк описывают каждая одну дорогу и содержат по три целых числа: u, v, w — соответственно, номера городов, которые соединены этой дорогой ($1 \le u \ne v \le n$) и максимальный вес груза, который можно перевозить по этой дороге ($1 \le w \le 10^9$). По дорогам разрешено движение в обе стороны, любые два города соединены не более чем одной дорогой.

Формат выходного файла

В выходной файл выведите слово `YES`, если указанный груз можно перевезти из города S в город T, и слово `NO` — в противном случае.

Примеры

path.in	path.out
4 4 5 1 2 5 2 3 6 2 4 4 3 4 7	YES
4 4 5 1 2 5 2 3 4 2 4 4 3 4 7	NO

Задача I. Две последовательности

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

Циклический преобразователь последовательностей (ЦПП) — это специальное устройство, работающее с последовательностями из чисел от 0 до $k-1$. Пусть задана последовательность a_1, a_2, $\ldots$, a_n. За одну операцию ЦПП может заменить a_i на $(a_i + d) \bmod k$ для всех $1 \le l \le i \le r \le n$. Такая операция в дальнейшем будет записываться как `l r d`.

Одной из интересных задач, которую может решать ЦПП, является преобразование одной последовательности в другую. Пусть заданы две последовательности a_1, a_2, $\ldots$, a_n и b_1, b_2, $\ldots$, b_n чисел от 0 до $k-1$. Необходимо преобразовать первую из них во вторую за не более чем $n-1$ операцию.

Формат входного файла

Первая строка входного файла содержит два целых числа n и k ($1 \le k < n \le 1000$). Вторая строка входного файла содержит n целых чисел a_1, a_2, $\ldots$, a_n. Третья строка входного файла содержит n целых чисел b_1, b_2, $\ldots$, b_n. Для всех a_i и b_i верно неравенство $0 \le a_i, b_i < k$.

Формат выходного файла

В первой строке выходного файла выведите количество преобразований c ($0 \le c < n$, его не требуется минимизировать). В последующих c строках выведите описания этих преобразований в формате `l r d`. Числа d для каждой из операций должны быть в диапазоне от нуля до $k-1$ включительно.

Примеры

twoseq.in	twoseq.out
5 3 0 0 1 2 2 1 1 2 0 0	3 1 2 1 3 3 1 4 5 1