Задача A. Озёра

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

Маленький котёнок живет на большом лугу. Недавно он решил построить себе новый дом, но никак не может выбрать подходящее место для него. На лугу расположены N маленьких озёр, в которых водится рыба. Каждый день котёнок ходит на рыбалку на одно из них. Для будущего дома он хочет выбрать такое место, чтобы расстояние от дома до самого дальнего озера было как можно меньше. Помогите ему в этом.

Так как луг очень большой, а котёнок и озёра маленькие, то можно считать луг бесконечной плоскостью, а дом и озёра — точками на ней. Более того, разрешается для дома выбрать точку, совпадающую с каким-либо из озёр. В таком случае котёнок построит дом на берегу этого озера.

Формат входного файла

В первой строке входного файла находится единственное число N — количество озёр на лугу ($1 \le N \le 10^3$). Далее в N строках находится по два целых числа x_i, y_i — координаты местонахождения озера. Координаты озёр не превосходят 10^4 по абсолютной величине. Местонахождения никаких двух озёр не совпадают.

Формат выходного файла

Выведите два числа — координаты точки, в которой котёнку надо построить свой дом. Если решений несколько, выведите любую такую точку. Координаты должны быть выведены с максимально возможной точностью, но не менее четырёх знаков после десятичной точки.

Пример

lakes.in	lakes.out
3 0 0 10 0 0 10	5.000000 5.000000

Примечания:

1. Решения, работающие при $N \le 3$, будут оцениваться исходя из 20 баллов.
2. Решения, работающие при $N \le 100$, будут оцениваться исходя из 60 баллов.

Задача B. Парковка

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

Во дворе дома проходит дорога, вдоль которой расположены парковочные места. Длина каждого из них составляет ровно один метр, а длина дороги — L метров. Среди жильцов дома есть N автовладельцев, и известно, что машина каждого из них занимает целое число парковочных мест в длину. Каждый рабочий день все они возвращаются домой в произвольном порядке и ставят машины на парковку, если есть такая возможность.

Автомобиль длины H можно поставить на парковку, начиная с места i ($1 \le i \le L$), если нет других машин, занимающих хотя бы одно парковочное место с номером от i до $min(i+H-1, L)$. То есть, машина не должна пересекаться ни с какой другой. Кроме того, автомобиль может выходить за границы парковки (в этом случае, передние или задние колёса окажутся на газоне). Однако, следует помнить, что машина должна находиться на парковке хотя бы какой-то своей частью, то есть занимать как минимум одно место на парковке.

Например, если $L = 5$, $H = 4$ и парковка пуста, то машину можно поставить начиная с 1 места (она займёт места 1 – 4), со 2 (места 2 – 5), с 3 места (займёт места 3 – 5, а также один метр газона), с 4 (займёт 2 метра газона) или с 5 места (займёт 3 метра газона). В случае, если место 4 уже занято машиной длины 1, то можно использовать только места, начиная с 5.

Автомобили приезжают в любом порядке и становятся на любое свободное место. Ваша задача — определить минимальное количество машин, которые можно расставить так, чтобы ни одна другая не поместилась на парковке. Известно, что длины всех машин различны.

Формат входного файла

В первой строке входного файла даны два целых числа — L и N. ($1 \le L \le 10^9$, $1 \le N \le 10^5$). Во второй строке даны длины машин — N различных целых чисел от 1 до L.

Формат выходного файла

В выходной файл выведите единственное число — минимальное количество машин необходимое для того, чтобы никакая другая не поместилась.

Примеры

park.in	park.out
10 4 1 4 3 2	3

Примечания:

В примере достаточно расставить машины с длинами 1, 2 и 3 метра (например, начиная с 1-ого, 5-ого и 10-ого парковочных мест. Последняя машина выходит за границы парковки на два метра). Ясно, что машину длиной 4 метра в данном случае уже некуда будет поставить.

Решение, работающее при $L \le 10^4, N \le 1000$ будет оцениваться исходя из 50 баллов.

Задача C. RGB...Z

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

Однажды Никита не знал, чем ему заняться на самом скучном уроке. Он начал рисовать в своей клетчатой тетрадке, закрашивая каждый кваратик некоторым цветом. С собой у него было много различных цветных гелиевых ручек. Его заинтересовало, можно ли раскрасить весь лист тетради ручками разных цветов так, чтобы ячейки с одинаковым цветом не были соседними (то есть, не имели общую сторону). При этом каждый цвет должен присутствовать в листке хотя бы один раз. Сам Никита не смог определить ответ на поставленный вопрос и поэтому обратился к соседу по парте, которым оказались Вы. Помогите справиться с задачей.

Для простоты будем считать, что листок бумаги имеет квадратную форму со сторонами длины N. Всего в нём $N \times N$ квадратиков.

Формат входного файла

В первой строке входного файла содержатся два числа N и K ($1 \le N \le 700$, $1 \le K \le min(26, N^2)$) — размер квадратного листа бумаги и количество ручек с различными цветами у Никиты.

Формат выходного файла

В первой строке выходного файла выведите "`YES`", если можно раскрасить листок требуемым образом и "`NO`" иначе. В случае положительного ответа в следующих N строках по N символов в каждой выведите какой-нибудь из требуемых способов покраски листка бумаги.

Цвета ручек Никиты обозначаются символами латинского алфавита: цвет первой ручки обозначается '`a`', второй — '`b`' и так далее.

Пример

rgbz.in	rgbz.out
2 4	YES ab cd
2 3	YES ab ca

Задача D. Камни

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

По кругу расположены N (N — четное) кучек камней, пронумерованных от 1 до N. Известно, сколько камней находится в каждой кучке. За один ход разрешается положить в две соседние кучки еще по несколько камней (одинаковое количество в каждую кучку).

Требуется последовательностью ходов уравнять количество камней во всех кучках или сообщить, что это невозможно. При этом длина последовательности ходов не должна превышать 1000.

Утверждается, что если существует какая-нибудь последовательность ходов, то существует также и такая, которая содержит не более 1000 ходов.

Формат входного файла

В первой строке входного файла находится единственное число N — количество кучек ($2 \le N \le 10^3$). N — четное. В следующей строке заданы N чисел — a_i ($1 \le a_i \le 100$) — количество камней в i-той кучке.

Формат выходного файла

Если уравнять кучки нельзя, то единственная строка выходного файла должна содержать фразу "`No solution`" без кавычек. Иначе первая строка должна содержать число p — количество ходов. Далее в p строках должны содержаться сами ходы. Ход описывается тремя числами: x_i, y_i, z_i, где x_i — номер первой кучки, y_i — номер второй кучки, z_i — число камней, дабавляемое в эти кучки. Номера x_i и y_i должны быть соседними (так как кучки расположены по кругу, то первая и последняя кучки являются соседними). z_i должно быть не менее 1 и не более 10^9.

Пример

stones.in	stones.out
4 1 1 3 3	1 1 2 2

Задача E. Вундеркинд

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

Вундеркинд Боря Вексер изучает английский. Сегодня он взял набор английских слов и решил составить из них составные слова таким образом, что каждое новое слово должно представлять собой конкатенацию ровно двух (необязательно разных) слов из исходного набора (конкатенация — это операция "склеивания" слов. Например, в результате конкатенации слов "table" и "spoon" могут получиться слова "tablespoon" и "spoontable").

После того, как Боря составляет новое слово, он записывает его в тетрадку. Конечно, это непростая работа, и Боря может ошибиться — перепутать порядок букв или выписать неправильные буквы. Он просит Вас проверить его.

Напишите программу, которая по набору исходных слов и набору слов, составленных Борей, определит, сколько слов Боря составил правильно, а в скольких из них перепутал местами буквы. Обратите внимание, что в некоторых словах Боря мог взять лишние буквы или, наоборот, пропустить нужные — такие слова учитывать не нужно.

Формат входного файла

В первой строке входного файла записаны два натуральных числа: N — количество слов в исходном наборе, и M — количество слов, составленных Борей ($1 \le N \le 100$, $1 \le M \le 1000$). В следующих N строках, по одному на строку, записаны слова исходного набора. Каждое исходное слово состоит не более чем из 101 буквы. Все исходные слова различны.

В следующих M строках, по одному на строку, записаны слова, составленные Борей. Каждое составленное слово состоит не более чем из 202 букв. Под словом подразумевается непустая последовательность строчных латинских букв.

Формат выходного файла

В первой строке выходного файла выведите количество правильно составленных Борей слов, то есть, представляющих собой конкатенацию ровно двух слов из исходного набора.

Во второй строке выведите количество слов из составленного Борей набора, которые могут быть получены из правильно составленных слов нетождественной перестановкой букв (такой, которая изменяет позицию хотя бы одной буквы).

Примеры

wunderkind.in	wunderkind.out
3 5 mean while borya mainwheel meanmean meanwhile oyabarmen boryavekser	2 2

Примечание

В вышеприведённом примере слова `meanwhile` и `meanmean` составлены правильно, а слова `mainwheel` и `oyabarmen` являются перестановками правильно составленных слов "meanwhile" и "meanborya".