Задача A. Равномерное распределение

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

Однажды Сергей собрался перелететь на воздушном шаре из Санкт-Петербурга в Рязань. Дорога предстояла дальняя. Для нормального полета на воздушном шаре требовалось содержать балласт, который располагался в n мешках прикрепленных к краю цилиндрической корзины.

Для равномерного распределения массы шара во всех мешках должно находиться одинаковое число камней. Однако, проверив все мешки, Сергей обнаружил, что в мешке с номером i находится a_i камней (для всех $i = 1 \ldots n$). За одно действие Сергей может переложить один камень из мешка в соседний с ним. Например, из второго мешка можно переложить в первый или третий, из первого — во второй или в мешок с номером n.

Помогите Сергею посчитать, за какое минимальное число действий можно равномерно распределить камни по мешкам, или понять, что этого достичь невозможно.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \le n \le 1000$). Вторая строка содержит n целых чисел a_i — количество камней в i-ом мешке ($0 \le a_i \le 10^6$).

Формат выходного файла

В выходной файл выведите, за какое минимальное число действий можно распределить камни данным образом. Если это сделать невозможно, выведите `-1`.

Примеры

balance.in	balance.out
2 1 3	1
3 1 2 3	1
3 1 2 2	-1
4 1 2 3 2	2

Задача B. Ключ

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

Гоша — начинающий компьютерный взломщик. После перехвата очередного сигнала между своими соседями, ему удалось извлечь из него два числа n и p. Гоша долго не мог понять смысл этих чисел, однако, в разговоре соседей на лестничной площадке нечаянно услышал алгоритм получения ключа, которого достаточно для полной расшифровки сигнала.

Из всех наборов натуральных чисел рассматриваются те, которые состоят из n элементов, а их произведение равно p. Ключ равен наибольшей из возможных сумм элементов такого набора.

Например, существует два набора из трех натуральных чисел, произведение которых равно четырем: $1, 2, 2$ и $1, 1, 4$. Сумма элементов первого набора равна пяти, а второго — шести, следовательно, ключ равен шести.

Помогите Гоше найти ключ для расшифровки сигнала.

Формат входного файла

Первая строка входного файла содержит два целых числа n и p ($1 \leq n, p \leq 10^3$).

Формат выходного файла

В выходной файл выведите искомый ключ.

Примеры

key.in	key.out
2 2	3
3 4	6

Задача C. Спички детям не игрушка!

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

По кругу расположены n кучек спичек, которые содержат $a_1, a_2, \ldots, a_n$ спичек. За одно действие Вася может взять две соседние кучки и объединить их в одну. Вася хочет добиться того, чтобы во всех получившихся кучках было одинаковое число спичек.

Он хочет узнать, какое минимальное число действий он должен совершить, чтобы добиться этой цели. Требуется написать программу, которая найдет оптимальную последовательность действий для Васи.

Формат входного файла

Первая строка входного файла содержит целое число n ($3 \le n \le 20000$). Вторая строка входного файла содержит n целых чисел a_1, a_2, …, a_n. Все a_i положительны, их сумма не превосходит 10^9.

Формат выходного файла

В первой строке выходного файла выведите число k — минимальное число действий, которое должен совершить Вася. Каждая из последующих k строк должна описывать одно действие и должна содержать два числа — номера объединяемых кучек. Изначально все кучки нумеруются натуральными числами от 1 до n. Каждая кучка, получаемая в результате объединения, получает минимальный еще не используемый номер. Так, например, кучка, получаемая в результате самого первого объединения, получит номер $n + 1$, в результате второго объединения — номер $n + 2$, и т. д.

Примеры

matches.in	matches.out
4 1 5 2 4	2 1 2 3 4
4 1 2 3 4	2 4 1 2 3
4 3 2 1 5	3 1 2 3 4 5 6

Задача D. Встретиться в середине

Имя входного файла:	`mitm.in`
Имя выходного файла:	`mitm.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

В стране K-мерике N городов и M дорог, соединяющих эти города. Каждая такая дорога соединяет два различных города, и движение по ней возможно в обе стороны. У дорог есть порядковые номера.

Шпион с кодовым именем Адам живет в городе A, а шпион с кодовым именем Бдам живет в городе B. Для передачи особо секретной информации им необходимо встретиться в середине дороги с номером X. Для этого они должны составить два маршрута, проходящих по дорогам и городам K-мерики: первый маршрут начинается в городе A, второй маршрут начинается в городе B, и оба маршрута оканчиваются в середине дороги X.

В целях конспирации, маршруты не должны пересекаться нигде, кроме как в середине дороги X. Более того, ни один маршрут не должен проходить через какой-либо город более одного раза.

Для того, чтобы правильно составить маршруты, шпионам необходимо учесть многое, в том числе и сколькими различными способами это можно сделать. Помогите шпионам и подсчитайте число различных пар маршрутов, удовлетворяющих изложенным требованиям. Так как это число может быть достаточно большим, выведите его по модулю P.

Формат входного файла

В первой строке входного файла находятся шесть целых чисел N ($2 \le N \le 20$), M ($1 \le M \le 200$), A ($1 \le A \le N$), B ($1 \le B \le N, A \ne B$), X ($1 \le X \le M$), P ($2 \le P \le 10^9$).

Далее в M строках находятся пары чисел S_i, F_i ($1 \le S_i, F_i \le N$). Каждая такая пара описывает одну дорогу, идущую из города с номером S_i в город с номером F_i. Дороги описаны в порядке возрастания номеров.

Пара городов может быть непосредственно соединена более чем одной дорогой. Никакая дорога не соединяет город с самим собой.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

mitm.in	mitm.out
2 1 1 2 1 10 1 2	1
4 5 1 4 3 100 1 2 1 3 2 3 2 4 3 4	2

Задача E. Раскраска кубиков

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

Недавно маленькому Вове подарили набор, состоящий из желтых кубиков. Вова долго играл с ними и, наконец, сложил из них прямоугольный параллелепипед размера $w \times h \times l$ кубиков.

Затем Вова решил, что данная конструкция будет гораздо красивее, если она будет красной, поэтому он покрасил ее. Несколько позже, разобрав ее, Вова заметил, что часть кубиков получилась окрашенной в красный только с одной стороны, у некоторых кубиков в красный были покрашены две грани и т.д.

Теперь маленького Вову интересует вопрос, сколько же кубиков имеют ровно k красных граней.

Формат входного файла

Во входном файле даны четыре целых числа — w, h, l и k соответственно ($1 \le w, h, l \le 100$, $0 \le k \le 6$).

Формат выходного файла

В выходной файл выведите одно число — количество кубиков с ровно k красными гранями.

Примеры

painting.in	painting.out
2 2 2 3	8
3 1 1 4	1

Задача F. Железные дороги

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

Россия Живет Дорогами.

Лозунг ОАО «РЖД»

Двумя основными системами, по которым устанавливается стоимость проезда на пригородных поездах являются зонная и покилометровая. При использовании зонной системы все станции разбиваются на так называемые «зоны», а стоимость проезда зависит от числа зон, которые проезжает пассажир. При использовании покилометровой системы стоимость проезда зависит от расстояния, которое проезжает пассажир.

Более формально, пусть на железной дороге есть вокзал и n станций, которые расположены вдоль одной прямой. В дальнейшем будем считать, что станции пронумерованы числами от 1 до n, а вокзал имеет номер 0. Расстояния от вокзала до i-ой станции обозначим как d_i, а номер зоны, в которой эта станции находится обозначим как z_i, при этом будем считать, что вокзал находится в зоне с номером 0.

Пусть необходимо вычислить стоимость проезда между станциями с номерами s и t. При расчете по зонной системе вычисляется абсолютная величина разности номеров зон $|z_s - z_t|$ и умножается на стоимость c_1 проезда одной зоны. При этом также необходимо учитывать, что если станции s и t находятся внутри одной зоны, то стоимость проезда между ними равна c_1. При расчете с использованием покилометровой системы вычисляется расстояние D между станциями. Это расстояние умножается на стоимость проезда одного километра: если $D < R$ (проезд на короткую дистанцию), то стоимость проезда одного километра составляет c_2, а иначе — c_3.

Ваша задача состоит в том, чтобы написать программу, которая вычисляет стоимость проезда по обеим системам для двух заданных станций.

Формат входного файла

Первая строка входного файла содержит целое число n ($3 \le n \le 50$). Вторая строка содержит n целых чисел $d_1, \ldots, d_n$ ($0 < d_1 < \ldots < d_n \le 1000$). Третья строка входного файла содержит n целых чисел $z_1, \ldots, z_n$ ($1 = z_1 \le \ldots \le z_n \le n$). Четвертая строка входного файла содержит два целых числа: s и t ($0 \le s, t \le n$, $s \ne t$). Пятая строка входного файла содержит четыре целых числа: c_1, c_2, c_3, R ($1 \le c_1, c_2, c_3, R \le 1000$);

Формат выходного файла

В первой строке выходного файла выведите стоимость, рассчитанную по зонной системе, а во второй — рассчитанную по покилометровой системе.

Примеры

railway.in	railway.out
3 3 4 5 1 2 3 1 3 5 2 3 4	10 4
3 5 6 7 1 1 1 1 0 1 2 3 5	1 15

Задача G. Дороги, дороги...

Имя входного файла:	`roads.in`
Имя выходного файла:	`roads.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

В одной из областей Флатландии находятся n городов, которые соединены между собой m дорогами, каждая из которых соединяет два города. Все дороги имеют асфальтовое покрытие, для поддержки которого в хорошем состоянии дороги необходимо постоянно ремонтировать.

Раньше этим занималась компания «ДорПром», однако теперь в рамках компании по борьбе с монополиями было решено часть дорог передать на обслуживание компании «Дороги Флатландии». Так как стоимость обслуживания дороги пропорциональна ее длине, то суммарная длина частей дорог, которые будут переданы на обслуживание компании «Дороги Флатландии», должна быть равна суммарной длине тех частей дорог, которые останутся на обслуживании у «ДорПрома». При этом допускается, чтобы разные части одной и той же дороги обслуживались разными компаниями.

Для простоты распределение дорог между компаниями было решено осуществить следующим образом. Проводится прямая линия, она делит область на две части. После этого все части дорог, которые оказываются по одну из сторон от прямой, передаются «Дорогам Флатландии», а по другую — «ДорПрому». Если же дорога оказывается в точности на этой прямой, то одна половина этой дороги передается «Дорогам Флатландии», а вторая — «ДорПрому».

Ваша задача состоит в том, чтобы написать программу, которая по описанию дорожной сети найдет такую прямую, что суммарные длины частей дорог, находящихся по разные стороны от нее, равны.

Формат входного файла

Первая строка входного файла содержит два целых числа: n ($2 \le n \le 100000$) и m ($1 \le m \le 100000$). Далее следуют n строк, описывающих города. Каждая из них содержит по два целых числа — координаты города, которые не превышают 10^3 по абсолютной величине. Координаты никаких двух городов не совпадают.

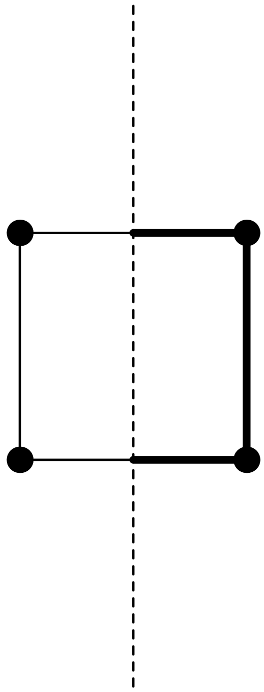
Далее следуют m строк, каждая из которых описывает одну дорогу и содержит два целых числа — номера городов, которые она соединяет. Гарантируется, что любые два города соединены не более, чем одной дорогой.

Формат выходного файла

В выходной файл выведите координаты двух различных точек, лежащих на найденной прямой. Первая строка должна содержать координаты одной из них, а вторая — другой. Эти координаты могут быть вещественными, но не должны превышать по абсолютной величине 10^3. Точки должны находиться друг от друга на расстоянии не менее чем 1. Выводите числа с максимальной возможной точностью. Равенство суммарных длин будет проверяться с точностью до 10^{-2}.

Примеры

roads.in	roads.out
4 4 0 0 0 1 1 1 1 0 1 2 2 3 3 4 4 1	0.5 0.0 0.5 1.0



На рисунке изображена дорожная сеть, соответствующая примеру, и искомая прямая. Более жирными линиями выделены части дорог, которые будут обслуживаться «Дорогами Флатландии», менее жирными — те части, которые будут обслуживаться «ДорПромом».

Задача H. Игра в слова

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

Петя и Вася очень любят играть в игры со словами. Недавно они придумали новую интересную игру «Найди слово». По ее правилам один из игроков заполняет клетки прямоугольника размером $n \times m$ буквами и пишет список слов. Второй игрок должен найти все слова из этого списка, которые встречаются в этом прямоугольнике. При этом слова могут быть записаны не только по горизонтали слева направо, но и по вертикали сверху вниз.

Пусть, например, первый игрок записал список слов idea, date, take и заполнил прямоугольник следующим образом:

i	d	e	a	x
t	a	k	d	y
a	t	b	c	z
d	e	e	f	w

Тогда idea и data встречаются в этом прямоугольнике (первое из них записано по горизонтали, а второе — по вертикали), а слово take не встречается.

В очередном раунде игры Петя заполнил прямоугольник буквами и записал список слов, которые должен найти Вася. Помогите Васе — напишите программу, которая для каждого из этих слов проверит, записано ли оно в составленном Петей прямоугольнике.

Формат входного файла

В первой строке входного файла два числа — n и m ($1 \le n, m \le 50$). Далее следуют n строк по m символов в каждой — результат содержимое прямоугольника, записанного Петей. Все символы в нем — строчные латинские буквы.

Следующая строка содержит одно целое число k ($1 \le k \le 50$) — размер списка. Далее следуют k строк, каждая из которых содержит по одному непустому слову длиной не более 50 из строчных латинских букв.

Формат выходного файла

В выходной файл выведите k строк. В каждой должно содержаться единственное слово «YES», если соответствующее слово встречается в прямоугольнике, и «NO» в противном случае.

Примеры

words.in	words.out
4 5 ideax takdy atbcz deefw 3 idea date take	YES YES NO

Задача I. Фигурное катание

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

В рамках зимних олимпийских игр во Флатландии проводятся соревнования по фигурному катанию. Для того, чтобы повысить объективность судейства, на этих соревнованиях применяется следующая система оценки.

Каждому из спортсменов оценки выставляют оценки n судей. Обозначим эти оценки как $a_1, \ldots, a_n$. Для получения итоговой оценки из этих оценок вычеркивается наибольшая и наименьшая, после чего вычисляется среднее арифметическое оставшихся оценок.

Ваша задача состоит в том, чтобы реализовать описанный алгоритм вычисления итоговой оценки.

Формат входного файла

Первая строка входного файла содержит целое число n ($3 \le n \le 100$). Вторая строка входного файла содержит n целых чисел: $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10$).

Формат выходного файла

В выходной файл выведите ответ на задачу с точностью не хуже 10^{-6}.

Примеры

skating.in	skating.out
4 1 2 3 4	2.5
4 3 3 3 3	3

Задача J. Вильгельм Телль

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

Вильгельм Телль — легендарный народный герой Швейцарии, искусный лучник, борец за независимость своей страны от Австрии и Священной Римской империи. Как известно, наместник германского императора в Швейцарии Геслер приказал Вильгельму сбить с головы собственного сына яблоко. Телль успешно справился с этой задачей, и теперь мы предлагаем вам повторить его подвиг.

Поскольку, яблоко и голова — слишком сложные геометрические объекты, то в этой задаче считается, что они представляют собой шары ненулевого радиуса. Тело ребенка, а также его руки и ноги не учитываются вовсе.

Стрела движется прямолинейно, кроме того задана позиция стрелка — точка, через которую обязана проходить траектория стрелы. Стрела пробивает яблоко насквозь, оставаясь при этом смертельно опасной.

Ваша задача — определить, можно ли выстрелить так, чтобы попасть в яблоко, но не в голову. При этом касание яблока считается попаданием, а касание головы за попадание не считается (легкая царапина, до свадьбы заживет).

Формат входного файла

Первая строка входного файла описывает яблоко и содержит два натуральных числа z и r ($z, r \le 100$). Это означает, что яблоко — шар с координатами $(0, 0, z)$ и радиусом r. Вторая строка входного файла содержит описание головы в аналогичном формате. Третья строка содержит координаты позиции стрелка — три целых числа x_0, y_0 и z_0 ($-100 \le x_0, y_0, z_0 \le 100$). Гарантируется, что расстояние от головы и яблока до этой точки не меньше 1.

Формат выходного файла

В случае если задача невыполнима, в единственной строке выходного файла выведите «`NO`». В противном случае выведите «`YES`» в первой строке выходного файла, а во второй — координаты точки на яблоке, в которую надо стрелять. Выводите ответ с максимальной точностью.

Примеры

wilhelmtell.in	wilhelmtell.out

`1 1`
`3 1`
`0 0 -1`
	`YES`
	`0.00000000 -0.86602540 0.50000000`

`1 1`
`6 4`
`0 0 -1`
	`NO`